Technische Universität Ilmenau
Fakultät für Informatik und Automatisierung
Fachgebiet Grafische Datenverarbeitung

Interaktive Auslegung von Kurvenscheibengetrieben

Diplomarbeit zur Erlangung des akademischen Grades
Diplominformatiker

Jens Storz
Matrikel-Nr.: 35048
Studiengang Informatik

Wintersemester 2007/2008

Betreuer: Dipl.-Inf. Ulf Döring (FG Grafische Datenverarbeitung)
Dipl.-Inf. Michael Reeßing (FG Konstruktionstechnik)
Verantwortlicher Hochschullehrer:
Prof. Dr. sc. techn. Beat Brüderlin (FG Grafische Datenverarbeitung)

Die Diplomarbeit wurde am 14.03.2008 bei der Fakultät für Informatik und Automatisierung der Technischen Universität Ilmenau eingereicht.

Inventarisierungsnummer: 2008-01-02/018/IN02/2252

Erklärung: „Hiermit versichere ich, dass ich diese Diplomarbeit selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe. Alle von mir aus anderen Veröffentlichungen übernommenen Passagen sind als solche gekennzeichnet.“

Ilmenau, 14.03.2008

Jens Storz

Inv.-Nr.: 2008-01-02/018/IN02/2252

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Aufgabenstellung

Die gezielte Auslegung von Kurvenscheiben stellt ein mächtiges Hilfsmittel zur Realisierung von Übertragungsfunktionen dar. Ziel dieser Arbeit ist es, ein Werkzeug zu konzipieren, welches Kurvenscheibengetriebe anhand einer editierbaren Übertragungsfunktion und wählbaren Strukturparametern generiert bzw. ändert. Basis für den Entwurf sollen VDI-Vorschriften und neuere in der Literatur beschriebene Ansätze bilden. Im Konzept muss das Beachten von Design-Constraints sowie das Synchronisieren von unterschiedlichen Übertragungsfunktionsrepräsentationen enthalten sein. Der Fokus der Arbeit soll auf dreigliedrigen ebenen Kurvenscheiben liegen.

1.2 Motivation

Es existiert eine Vielzahl von Projekten, die sich u.a. mit Bewegungsdesign oder Kurvenscheibensynthese befassen. Beispielhaft für solche Projekte seien hier GENESYS [9] und CADiS [8] genannt.
Auf den Großteil der erhältlichen Programme trifft leider mindestens eine der folgenden Eigenschaften zu, wodurch eine eigene Entwicklung gerechtfertigt ist:

- keine freie Verfügbarkeit

- starke Abhängigkeit von spezieller oder veralteter Software
- komplizierte Handhabung
- an eigene Bedürfnisse schwer anpassbar
- unzureichende Funktionalität

Die Digitale Mechanismen- und Getriebebibliothek (Kurzbezeichnung: DMG-Lib) [4] hat sich u.a. zum Ziel erklärt, das Wissen um Mechanismen und Getriebe aufzubereiten und öffentlich zugänglich zu machen. In diesem Kontext wurde bespielsweise auch MASP (Modelling and Analysis of Solution Principles) [12] in die DMG-Lib aufgenommen und in diesem Rahmen weiter entwickelt. MASP ist ein Programm, mit dem es z.B. möglich ist, Getriebe aus einzelnen Elementen zu konstruieren, deren Verhalten über Randbedingungen zu definieren und zu analysieren. Auch Kurvenscheibengetriebe können als Elemente von Getrieben angesehen werden. Jedoch ist das Erstellen von Kurvenscheibengetrieben ein komplexer Vorgang, was ein spezielles Werkzeug notwendig macht. Unter Zuhilfenahme eines solchen Werkzeugs können Kurvenscheibengetriebe generiert und anschließend in MASP integriert werden, wo sie sich mit anderen Getrieben koppeln lassen und das Gesamtverhalten analysiert werden kann.

1.3 Einschränkung

Der in dieser Arbeit beschriebene Editor für die Auslegung von Kurvenscheibengetrieben berechnet Stützpunkte einer Kurvenscheibe auf Basis geometrischer Zusammenhänge. Er beinhaltet keine Betrachtung von physikalischen Zusammenhängen, wie Kräfte, Reibung oder Material. Die berechnete Kurvenscheibe ermöglicht dem Abtriebsglied bei ständigem Kontakt, die von der Übertragungsfunktion geforderte Position einzunehmen. Die Zwanglaufsicherung des Abtriebsglieds ist damit allein jedoch nicht sichergestellt.

1.4 Aufbau der Arbeit

Die Arbeit ist wie folgt gegliedert. Kapitel 2 beschreibt die Grundlagen zur Definition einer Übertragungsfunktion und zur Auslegung von Kurvenscheibengetrieben. Ist eine Übertragungsfunktion definiert und das Getriebe parametrisiert, kann die Kurvenscheibenkontur ermittelt werden. Auch hierzu werden die Grundlagen in diesem Kapitel erläutert. Außerdem beschreibt dieses Kapitel Ansätze, die Alternativen zu den VDI-Empfehlungen aufzeigen, die im weiteren Verlauf der Arbeit genutzt werden. Kapitel 3 beschäftigt sich mit den Möglichkeiten der Interaktion des Nutzers zur Bearbeitung von Übertragungsfunktion und Getriebeparametern, einschließlich der Visualisierungsmöglichkeiten. Das beinhaltet auch eine Betrachtung der Validierung der erzeugten Kurvenscheiben. Im Anschluss (Kapitel 4), werden Hinweise zur softwaretechnischen Umsetzung der Konzepte gegeben. In Kapitel 5 wird die Arbeit zusammengefasst und ein kurzer Ausblick auf zukünftige Aufgaben gegeben. Im Anhang finden sich Berechnungen (Anhang A) zur Koeffizientenermittlung nicht normierter Funktionsabschnitte sowie Beispiele (Anhang B) zur Anwendung des Kurvenscheibeneditors und zugehöriger Datenhaltung.

Kapitel 2

Grundlagen

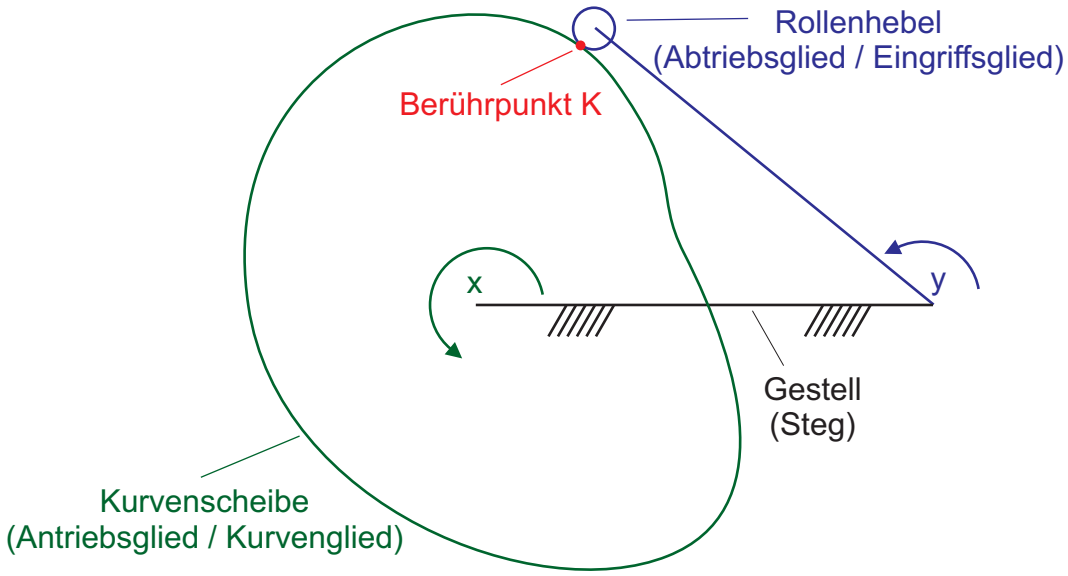


Abbildung 2.1: *Rollenhebel auf Außenflanke einer Kurvenscheibe*

Ein dreigliedriges Kurvengetriebe besteht aus einem Kurvenglied und einem Eingriffsglied, welche durch einen Steg verbunden sind. Weiterhin berühren sich Kurvenglied und Eingriffsglied im Berührpunkt K. Das Kurvenglied wird durch eine Dreh- oder Schubbewegung angetrieben und verändert so, wiederum durch Dreh- oder Schubbewegungen, die Lage des Eingriffsgliedes. Wie diese Lageveränderung vollführt wird, ist durch die Übertragungsfunktion des Getriebes beschrieben. Der Schwerpunkt dieser Arbeit liegt auf zweidimensionalen Kurvengetrieben, deren Kurvenglied über eine

Rotationsbewegung angetrieben wird (siehe Abb. 2.1). In diesem Fall wird das Kurvenglied als Kurvenscheibe bezeichnet. Der Begriff Abtriebsglied ist in dieser Arbeit als Synonym für den Begriff Eingriffsglied anzusehen.
Nach der VDI-Richtlinie 2142 Blatt 1 [18, S. 2] umfasst der Ablauf zur Herstellung eines Kurvengetriebes folgende Schritte:

1. Aufstellen eines Bewegungsplans
2. Aufbau des Bewegungsdiagramms
3. Auswahl des Getriebetyps
4. Ermittlung der kinematisch-geometrischen Hauptabmessungen
5. Berechnung des Kurvenprofils und ggf. weiterer Größen
6. Konstruktion des Getriebes
7. Anfertigen der Zeichnung für das Kurvenglied und ggf. weiterer Bauteile
8. Fertigung des Getriebes

Ein Bewegungsplan gibt Forderungen an die Übertragungsfunktion vor. Werden diese Forderungen in einem Editor für die Übertragungsfunktion eingegeben, kann daraus ein Bewegungsdiagramm erstellt werden, in welchem der exakte Funktionsverlauf beschrieben wird. In einem Editor für die Getriebeauslegung können dann unter Einbeziehung der Übertragungsfunktion die Punkte drei bis fünf durchgeführt werden. Dadurch werden die Grundlagen geschaffen, um die Punkte sechs bis acht durchführen zu können.
Der Ablauf zur Erstellung eines Kurvenscheibengetriebes unter Verwendung des in dieser Arbeit beschriebenen Editors wird vereinfacht im Schema in Abbildung 2.2 dargestellt.

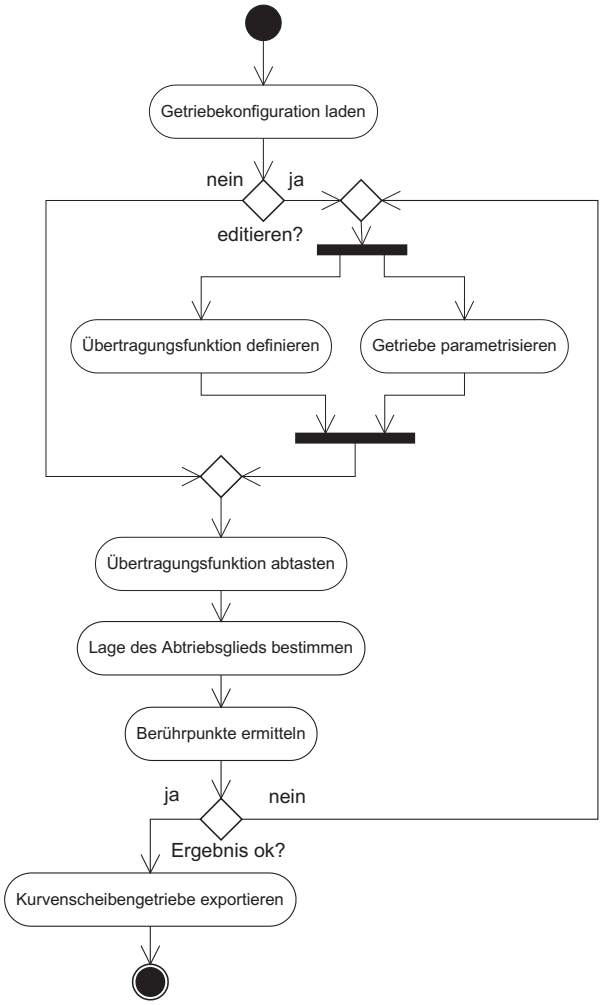


Abbildung 2.2: *vereinfachte Darstellung des Verfahrens zur Erstellung von Kurvenscheibengetrieben*

2.1 Übertragungsfunktion

Die Übertragungsfunktion f eines Kurvenscheibengetriebes gibt die Position bzw. den Winkel y des Abtriebsglieds in Abhängigkeit vom wiederum zeitabhängigen Winkel x des Antriebsgliedes an ($y = f(x(t))$). Benötigt eine Kurvenscheibe für eine Umdrehung die Zeit T, so ist leicht ersichtlich, dass $x(n \cdot T + t) = x(m \cdot T + t)$ für alle Umdrehungen n, m ($n, m \in N$) zu jedem beliebigen Zeitpunkt t gelten muss, um jederzeit die Übertragungsfunktion erfüllen zu können. Das impliziert, dass die Übertragungsfunktion z.B. bei Ein- und Ausschaltvorgängen des Antriebs nicht erfüllt werden kann, da die

Scheibe ggf. zunächst eine Sollgeschwindigkeit erreichen muss. Im Folgenden wird zur Vereinfachung der Betrachtungen von einer konstanten Antriebswinkelgeschwindigkeit ausgegangen, womit x von t linear abhängig ist und $x = (k \cdot t) \mod 2 \cdot \pi$ gilt. Über k kann dann die Zeitdauer beeinflusst werden, die eine Umdrehung der Antriebskurvenscheibe benötigt. Durch diesen direkten Zusammenhang wird in dieser Arbeit für die erste Ableitung nach dem Antriebswinkel auch der Begriff Geschwindigkeit verwendet, ebenso für die zweite Ableitung der Begriff Beschleunigung.

2.1.1 Funktionsabschnitte

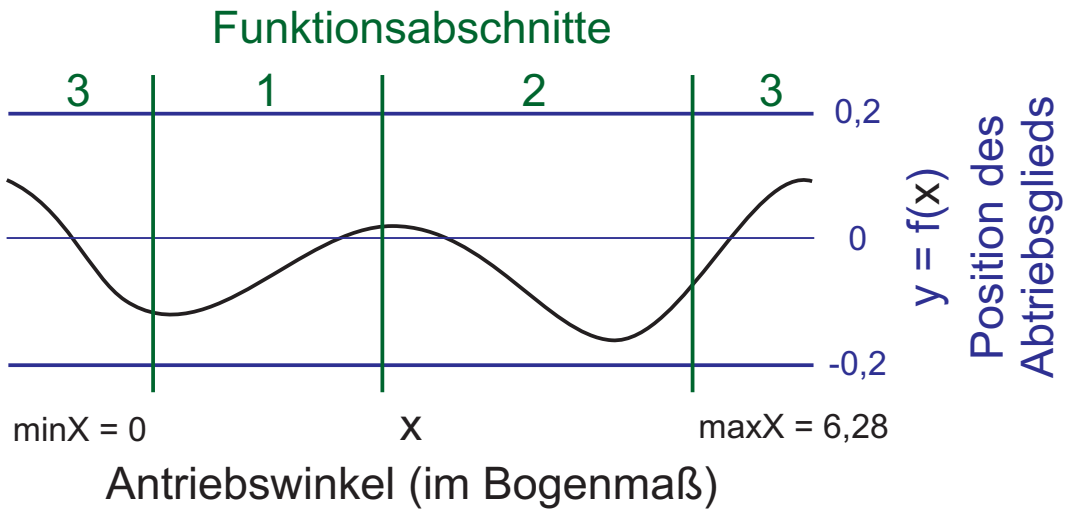


Abbildung 2.3: *Abschnitte einer Übertragungsfunktion*

Die Übertragungsfunktion, die auch als Bewegungsdiagramm bezeichnet werden kann, entsteht aus einem geforderten Bewegungsplan (s. Abb. B.4). Um die dadurch entstehende variable Komplexität der Übertragungsfunktion handhabbar zu machen, wird die Übertragungsfunktion aus Teilfunktionen (s. Abb. 2.3) zusammengesetzt. Dabei werden die Positionen der Abschnittsgrenzen sowie die Werte für Geschwindigkeit (1. Ableitung) und Beschleunigung (2. Ableitung) an dieser Stelle vorgegeben. Da die Übertragungsfunktion periodisch ist und es nicht notwendig ist, eine Grenze bei x_{min} zu definieren, setzt sich der letzte Abschnitt am Beginn der Funktionsdefinition fort. Nach der VDI-Richtlinie 2143 [16] werden die Grenzen eines Abschnittes in

	Rast	konst. Geschw.	Umkehr	Bewegung
$v = y'$	0	$\neq 0$	0	$\neq 0$
$a = y''$	0	0	$\neq 0$	$\neq 0$

Tabelle 2.1: *Systematik der Bewegungsaufgaben*

vier Kategorien unterteilt (s. Tab. 2.1). Da jeder Abschnitt zwei Grenzen hat, gilt es 16 Kombinationen von Abschnittsgrenzen zu unterscheiden und dementsprechend die Funktionsverläufe der Funktionsabschnitte zu beschreiben.
Alternativ zu der Zerlegung der Funktion in Abschnitte, lässt sich eine Funktion auch durch Überlagerung harmonischer Anteile ausdrücken. Darauf soll in dieser Arbeit jedoch nicht näher eingegangen werden.

Verlaufstypen

Von einer einfachen Gerade über Winkelfunktionen bis hin zu Polynomen höheren Grades, ist eine Vielzahl möglicher Funktionstypen denkbar, die einen einzelnen Funktionsabschnitt mehr oder weniger gut an die Anforderungen angepasst beschreiben können. In Blatt 2 der VDI 2143 [17, S. 13] findet sich eine Tabelle, die einige mögliche Funktionstypen für die 16 Kombinationen von Abschnittsgrenzen aufzeigt. Hier fällt besonders auf, dass ein Polynom 5. Grades für jede dieser Kombinationen genutzt werden kann und somit universell einsetzbar ist (s. Abschnitt 2.1.2: Polynom 5. Grades als Universallösung).

Normierung der Funktionen

In der einschlägigen Literatur [16] [19] wird die Verwendung von normierten Übertragungsfunktionen für die einzelnen Funktionsabschnitte empfohlen, d.h. ein Funktionsabschnitt ist definiert als $f(x)$ mit $0 \leq x \leq 1$, $f(0) = 0$ und $f(1) = 1$. Diese Einschränkung vereinfacht die Berechnung der Übertragungsfunktion und ermöglicht die Verwendung allgemeingültiger Wertetabellen. Nach erfolgter Ermittlung der normierten Übertragungsfunktion muss die Funktion auf die realen Grenzwerte gestreckt

werden.

Stoß- und Ruckfreiheit

Eine häufige Forderung an Übertragungsfunktionen ist Stoß- und Ruckfreiheit, d.h. die Funktion soll keine Sprünge im Verlauf ihrer ersten und zweiten Ableitung aufweisen. Ein Stoß stellt einen Geschwindigkeitssprung dar, was eine theoretisch unendlich große Beschleunigung erfordert. Ein Ruck wiederum stellt einen Beschleunigungssprung dar. Somit ist klar, dass zunächst die Freiheit von Stößen höhere Priorität genießt. Bei der Zusammensetzung der Übertragungsfunktion aus Teilfunktionen muss auf Erfüllung dieser Forderung insbesondere an den Abschnittsgrenzen geachtet werden, d.h. sowohl der Funktionswert als auch die ersten beiden Ableitungen an der Stelle müssen für die beiden angrenzenden Funktionsabschnitte identisch sein. Nicht zuletzt aus diesem Grund sollten bei der Definition einer Übertragungsfunktion für Kurvenscheibengetriebe diese drei Funktionsverläufe stets synchron betrachtet werden (s. Abb. 2.4).

2.1.2 Abweichungen von der VDI-Richtlinie

In diesem Abschnitt wird eine von den Empfehlungen der VDI-Richtlinie [16] abweichende Verfahrensweise vorgestellt. Dabei wird vor allem Wert auf Übersichtlichkeit und Nachvollziehbarkeit gelegt.

Verzicht auf Systematik der Bewegungsaufgaben

Auf eine Unterteilung der Abschnittsgrenzen, wie in Tabelle 2.1 beschrieben, kann durchaus verzichtet werden. Die Berechnungsvorschriften einiger Bewegungsgesetze für den Kurvenverlauf eines Abschnitts (z.B. Polynom 5. Grades) können so gewählt werden, dass sie für jede der 16 Kombinationen von Grenztypen identisch ist. Dabei wird jeder Funktionsabschnitt wie ein allgemeiner Bewegungsabschnitt behandelt.

Verzicht auf normierte Übertragungsfunktionen

Der Vorteil der Nutzung normierter Übertragungsfunktionen liegt in einer einfacheren Koeffizientenermittlung. Die Nutzung allgemeingültiger Wertetabellen ist jedoch

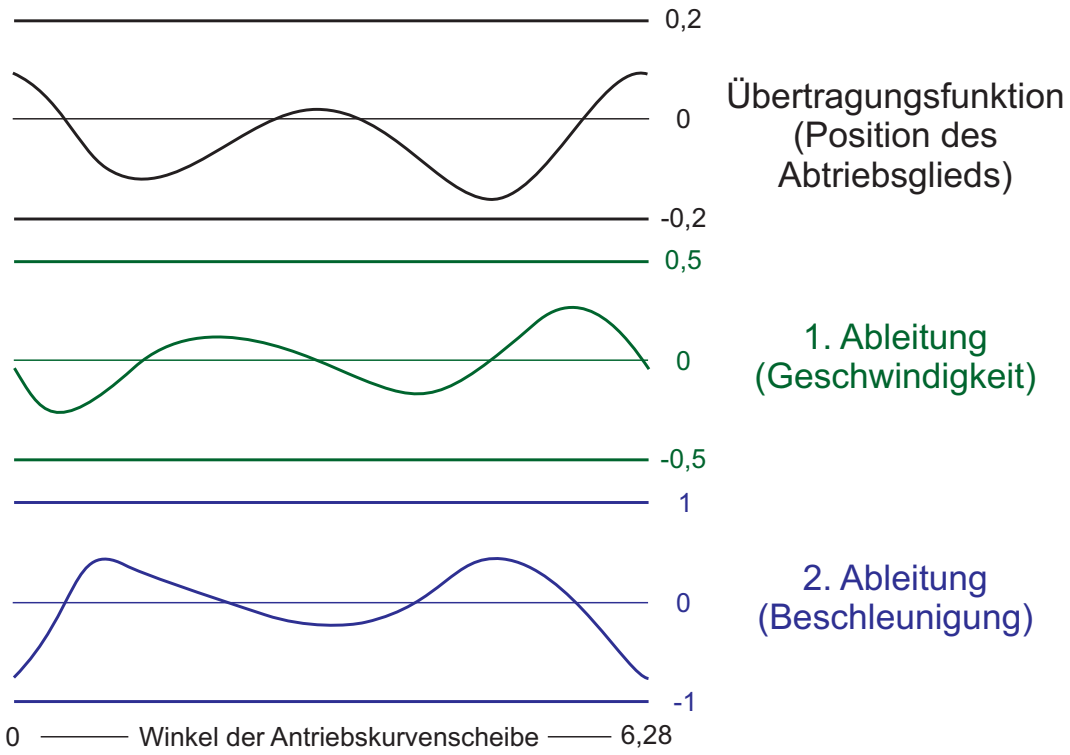


Abbildung 2.4: *Übertragungsfunktion und ihre Ableitungen für eine Umdrehung einer Kurvenscheibe*

bei Computerunterstützung höchstens zur Validierung der Berechnungen interessant. Es gibt aber sogar mehrere Gründe, die gegen eine Verwendung normierter Übertragungsfunktionen sprechen. Zunächst einmal kann die Streckung Sonderfälle abverlangen; nämlich genau dann, wenn die Funktionswerte an den Abschnittsgrenzen identisch sind (s. Abb. 2.5). Die realen Funktionswerte werden u.a. bei allen Berechnungen der Kurvenscheibensynthese und für die Ermittlung jedes Stützpunkts benötigt. Um die Editierbarkeit zu gewährleisten muss eine möglichst hohe Zahl an Stützpunkten in Echtzeit berechnet werden können, wobei die häufige Umrechnung von normierten in reale Koordinaten als unnötiger Aufwand anzusehen ist. Die Koeffizientenermittlung nicht normierter Funktionen ist zwar aufwändiger, jedoch gilt es zu beachten, dass die Lösung der Gleichungssysteme zur Koeffizientenermittlung bereits ins Programm integriert werden kann und nicht erst zur Laufzeit erfolgen muss. Weiterhin ist eine Neuberechnung des Funktionsverlaufs nur nötig, wenn der jeweilige Funktionsabschnitt

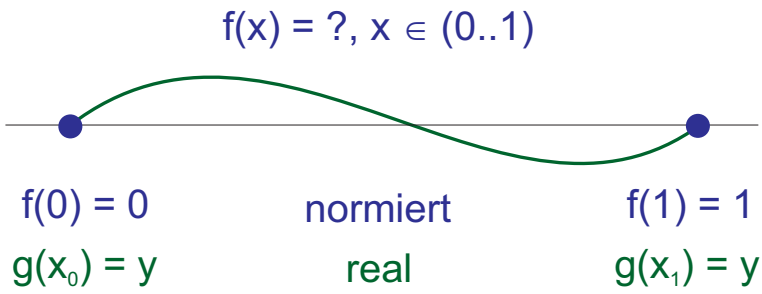


Abbildung 2.5: *Sonderfall bei normierten Übertragungsfunktionen. Dargestellt ist der real gewünschte Funktionsverlauf. Der Verlauf der normierten Übertragungsfunktion ist unklar.*

editiert wird.

Polynom 5. Grades als Universallösung

Der Einsatz einer großen Zahl von Funktionsabschnittstypen ist nicht zwingend sinnvoll. Nach Braune [3] ist die alleinige Verwendung von Polynomen 5. Grades in aller Regel ausreichend. Zwar kann ein anderer Funktionstyp beispielsweise bessere Maximalwerte bzgl. Geschwindigkeit oder Beschleunigung aufweisen, doch selbst diesem Umstand kann man leicht begegnen, indem man den Abschnitt in mehrere Teilabschnitte zerlegt, um das verlangte Verhalten exakter nachzubilden. Durch den Verzicht auf eine Unterteilung nach Tab. 2.1 sind jedoch zumindest noch zwei weitere Funktionen unerlässlich: Das Polynom 0. sowie 1. Grades. Damit kann eine schlichte Rast bzw. eine konstante Geschwindigkeit des Abtriebsgliedes über einen Funktionsabschnitt realisiert werden. Zwar können diese Polynome auch durch ein Polynom 5. Grades ausgedrückt werden, doch die explizite Deklaration als Polynom niederen Grades erleichtert das Editieren enorm und verhindert ein versehentliches Ändern z.B. einer Gerade in eine beliebige Bewegung. Der Implementierungsaufwand für diese Funktionen ist minimal.

2.1.3 Verlaufsberechnung eines Funktionsabschnitts

Hier soll anhand von Beispielen die Berechnung des Verlaufs eines Funktionsabschnitts erklärt werden. Für Polynome müssen dazu die Koeffizienten ermittelt werden. Außer-

dem wird auf die Anpassung der Randwerte der Teilfunktionen eingegangen, wodurch Stoß- und Ruckfreiheit realisiert werden kann.

Polynom 0. Grades

Die einfachste Möglichkeit eines Funktionsverlaufs stellt das Polynom 0. Grades dar. Es hat die Form $f(x) = k$, wobei k eine Konstante ist. Der Funktionsabschnitt ist zwischen den Punkten s_{min} und s_{max} gültig und stellt eine Rast dar. Geschwindigkeit und Beschleunigung sind an jeder Stelle 0.

Polynom 1. Grades

Das Polynom 1. Grades hat die Form $f(x) = k_1 \cdot x + k_0$. Die erste Ableitung hat an jeder Stelle den Wert k_1 (konstante Geschwindigkeit), die zweite Ableitung an jeder Stelle den Wert 0. Die zwei Koeffizienten k_1 und k_0 lassen zwei Bedingungen in einem Gleichungssystem zu. Hierfür werden die Funktionswerte an den Grenzen des Funktionsabschnitts eingesetzt:
Seien $x_0 = x_{min}$, $y_0 = f(x_{min})$, $x_1 = x_{max}$, $y_1 = f(x_{max})$ gegeben.

$$y_0 = k_1 \cdot x_0 + k_0 \tag{2.1}$$

$$y_1 = k_1 \cdot x_1 + k_0 \tag{2.2}$$

Daraus ergibt sich k_1 zu $(y_1 - y_0)/(x_1 - x_0)$ und k_0 zu $y_0 - k_1 \cdot x_0$. Weiter ergibt sich die Bedingung, dass x_{min} und x_{max} nicht denselben Wert aufweisen dürfen.

Polynom 3. Grades

Bisher sind in die Berechnungen die Ableitungen der Funktion nicht eingeflossen. Das war aufgrund des niedrigen Grades der Polynome auch nicht möglich. Mit einem Polynom 3. Grades ($f(x) = k_3 \cdot x^3 + k_2 \cdot x^2 + k_1 \cdot x + k_0$) besteht die Möglichkeit, vier Bedingungen anzugeben, die für den gewünschten Verlauf zweckmäßig sind. In diesem Fall sind durch den Nutzer neben den Funktionswerten auch die Werte für die 1. Ableitung (Geschwindigkeit des Abtriebsglieds) an den Funktionsabschnittsgrenzen vorzugeben.

Die Werte für die Geschwindigkeit werden hier mit v_0 bzw. v_1 bezeichnet und sind dann entsprechend in die Ableitung des Polynoms ($f'(x) = 3 \cdot k_3 \cdot x^2 + 2 \cdot k_2 \cdot x + k_1$) einzusetzen.

$$y_0 = k_3 \cdot x_0^3 + k_2 \cdot x_0^2 + k_1 \cdot x_0 + k_0 \tag{2.3}$$

$$y_1 = k_3 \cdot x_1^3 + k_2 \cdot x_1^2 + k_1 \cdot x_1 + k_0 \tag{2.4}$$

$$v_0 = 3 \cdot k_3 \cdot x_0^2 + 2 \cdot k_2 \cdot x_0 + k_1 \tag{2.5}$$

$$v_1 = 3 \cdot k_3 \cdot x_1^2 + 2 \cdot k_2 \cdot x_1 + k_1 \tag{2.6}$$

Durch das Lösen des Gleichungssystems erhält man die Koeffizienten für das Polynom. Eine Übertragungsfunktion, die ausschließlich über diese Polynome 3. Grades definiert wurde, ist dann stoßfrei. Ruckfreiheit kann nur garantiert werden, wenn sich die benachbarten Funktionsabschnitte an die Randwerte der 2. Ableitung (Geschwindigkeit) des Polynoms anpassen können. Dies ist beispielsweise mit einem Polynom 5. Grades möglich.

Polynom 5. Grades

Ein Polynom 5. Grades hat die allgemeine Form $f(x) = k_5 \cdot x^5 + k_4 \cdot x^4 + k_3 \cdot x^3 + k_2 \cdot x^2 + k_1 \cdot x + k_0$. Durch die sechs Koeffizienten können in einem Gleichungssystem sechs Bedingungen einfließen, die das Polynom bestimmen. Diese Bedingungen sind zweckmäßig $f(x_0) = y_0$, $f'(x_0) = v_0$, $f''(x_0) = a_0$, $f(x_1) = y_1$, $f'(x_1) = v_1$, $f''(x_1) = a_1$. Wie die Koeffizienten des Polynoms aus den Bedingungen mittels Derive [13] ermittelt werden können, wird im Anhang A.1 gezeigt. Das entstandene Polynom 5. Grades kann als Bindeglied zwischen beliebigen Funktionen fungieren und dabei Stoß- und Ruckfreiheit garantieren. Natürlich ist auch eine Funktionsbeschreibung möglich, deren Abschnitte ausschließlich aus Polynomen 5. Grades besteht.

Weitere Funktionstypen

Polynome mit einem Grad größer 5 bieten zusätzliche Möglichkeiten, da weitere Bedingungen zur Koeffizientenbestimmung hinzugezogen werden können. Denkbar wäre

die Festlegung der Randwerte der 3. Ableitung (Ruckfunktion) oder aber Funktionswerte zwischen den Abschnittsgrenzen zu definieren. Auf diese Funktionstypen wie auch auf die Beschreibung von Funktionsabschnitten unter Verwendung trigonometrischer Funktionen soll hier nicht näher eingegangen werden. Die einschlägige Literatur [16] [19] bietet hier weiterführende Informationen.

2.1.4 Zusammenfügen von Funktionsabschnitten

Wenn davon ausgegangen werden kann, dass ein Funktionsabschnitt nach Wahl des Verlaufstyps (z.B. Polynom 5. Grades) höchstens über den Funktionswert (y) sowie Geschwindigkeit (v) und Beschleunigung (a) an den Abschnittsgrenzen parametrisiert wird, so kann ein entsprechender Datentyp wie folgt definiert werden: Die Übertragungsfunktion besteht aus einer geordneten Liste, deren Elemente einen Abschnitt beschreiben, indem sie den Verlaufstyp, den Beginn des Abschnitts (nachdem die Liste geordnet wird) sowie die Werte für y, v und a an dieser Stelle enthalten. Für die Berechnung des Verlaufs wird auch das Ende des Abschnitts benötigt, dieser entspricht dem Abschnittsbeginn des Folgeabschnitts. Dadurch wird gewährleistet, dass die Funktion zu jedem Zeitpunkt der Bearbeitung vollständig definiert ist.
Da die Funktion geschlossen ist und nicht mit einer Abschnittsgrenze beginnen muss, taucht einmalig der Fall auf, dass das Ende eines Funktionsabschnitts noch vor dessen Anfang liegt. Hier muss auf die Endposition der Maximalwert weniger dem Minmalwert addiert werden. $x_{1,neu} = x_{1,alt} + maxX - minX$, wobei gewöhnlich gilt: $maxX = 2 \cdot \pi$ und $minX = 0$.
Nicht alle Verlaufstypen nutzen alle angegebenen Parameter. Dennoch müssen alle Parameter angegeben werden. Zum einen könnte der vorangehende Abschnitt diese Parameter benötigen, zum anderen lässt sich der Verlaufstyp so ohne Umstände jederzeit ändern.

2.1.5 Automatische Parameterkorrektur

Beim Zusammenfügen von Funktionsabschnitten ist es wichtig, dass der Übergang stetig und differenzierbar ist. Um dies zu gewährleisten kann eine automatische Parame-

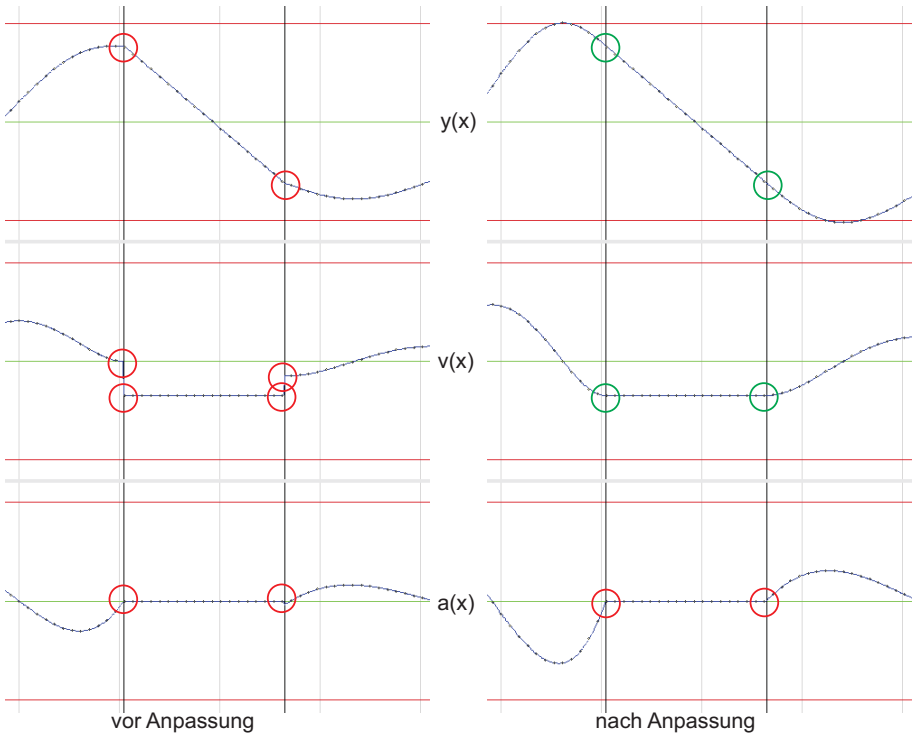


Abbildung 2.6: *Anpassung eines Polynoms 5. Grades an ein Polynom 1. Grades*

terkorrektur genutzt werden. In dieser Arbeit wurden dabei zwei Fälle unterschieden. Im ersten Fall liegt ein Polynom niederen Grades vor, welches nicht alle Parameter nutzt. Dadurch kann es zu unerwünschten Sprüngen oder Knicken im Funktionsverlauf oder dessen Ableitungen kommen. Durch Veränderung des vom Polynom niederen Grades ungenutzten Parameters, kann sich ein benachbartes Polynom höheren Grades an den Verlauf anpassen. In Abb. 2.6 werden die Auswirkungen gezeigt, wenn man Polynome 5. Grades an ein Polynom 1. Grades anpasst. In der Funktionsdefinition sind zunächst Knicke an den Abschnittsgrenzen zu sehen und in der Geschwindigkeitsfunktion Sprünge. Indem die Parameter für die 1. Ableitung der Polynome 5. Grades angeglichen werden, werden die Verläufe geglättet. Es ist zu beachten, dass es mit einem Polynom 5. Grades in diesem Fall nicht möglich ist, auch die 2. Ableitung differenzierbar zu machen. Dies wäre beispielsweise mit einem Polynom 7. Grades möglich, wenn als zusätzliche Bedingungen $f'''(x_0) = 0$ und $f'''(x_1) = 0$ gefordert werden.
Im zweiten Fall sind Polynome 5. Grades benachbart. Hier kann auch die 2. Ableitung

(Beschleunigungsverlauf) geglättet werden. Um dies zu erreichen, muss sichergestellt werden, dass auch die 3. Ableitung (Ruckfunktion) an der Schnittstelle benachbarter Abschnitte identisch ist. Braune [3] beschreibt, wie diese Knicke durch Korrektur der Parameter für die zweite Ableitung entfernt werden können. Dabei wird davon ausgegangen, dass der Wert der 2. Ableitung an der Schnittstelle unter Beachtung weiterer Forderungen ggf. weniger relevant ist, als deren Differenzierbarkeit. Ist der Anstieg der 2. Ableitung des linken Funktionsabschnitts größer als der des rechten, so kann eine Glättung an der Schnittstelle erreicht werden, indem der Wert für die 2. Ableitung an dieser Stelle gesenkt wird. Damit ändert sich ein Parameter, der zur Berechnung beider anliegenden Funktionsabschnitte benötigt wird. Dieser Parameter hat wiederum Einfluss auf alle sechs Koeffizienten eines Polynoms 5. Grades. Da das Polynom 5. Grades durch die Werte y, v und a an dessen Abschnittsgrenzen (x_0 und x_1) bestimmt wird, verändert sich auch der Wert der 3. Ableitung an der einen Grenze, sobald der Wert für die 2. Ableitung an der anderen Grenze verändert wird. Ist also eine Glättung an der einen Grenze erfolgt, können an den benachbarten Grenzen neue Knicke entstehen. Um also die 2. Ableitung an jeder Abschnittsgrenze zu glätten, müssten alle Abschnitte in die Rechnung aufgenommen werden. Da die Anzahl der Abschnitte aber erst während der Bearbeitung bekannt wird, ist dies ein recht schwieriges Problem. Hinzu kommt, dass keine eindeutige Lösung existiert, da eine Grenze nicht nur durch den Parameter an dieser Stelle, sondern auch durch die Parameter an den benachbarten Grenzen geglättet werden kann.

Eine einfache und dennoch sinnvolle und effiziente Alternative stellt folgendes iteratives Verfahren dar. Man betrachte die Differenz der 3. Ableitung (Ruckfunktion) an jeder Grenze und verschiebe den Parameter für die 2. Ableitung ein kleines Stück in die entsprechende Richtung. Diesen Prozess wiederholt man, bis die Differenz der 3. Ableitung an jeder Abschnittsgrenze einen Sollwert unterschreitet. Auf diese Weise wird sichergestellt, dass die Veränderung an der einen Grenze keine zu großen Auswirkungen an den Nachbargrenzen hat. Außerdem werden alle Grenzen gleichermaßen erfasst und geglättet (s. Abb. 2.7).

Bei Nutzung dieses Verfahrens ist zu beachten, dass der gesamte Kurvenverlauf beeinflusst wird und dabei ggf. auch zulässige Grenzen überschritten werden können. Es ist

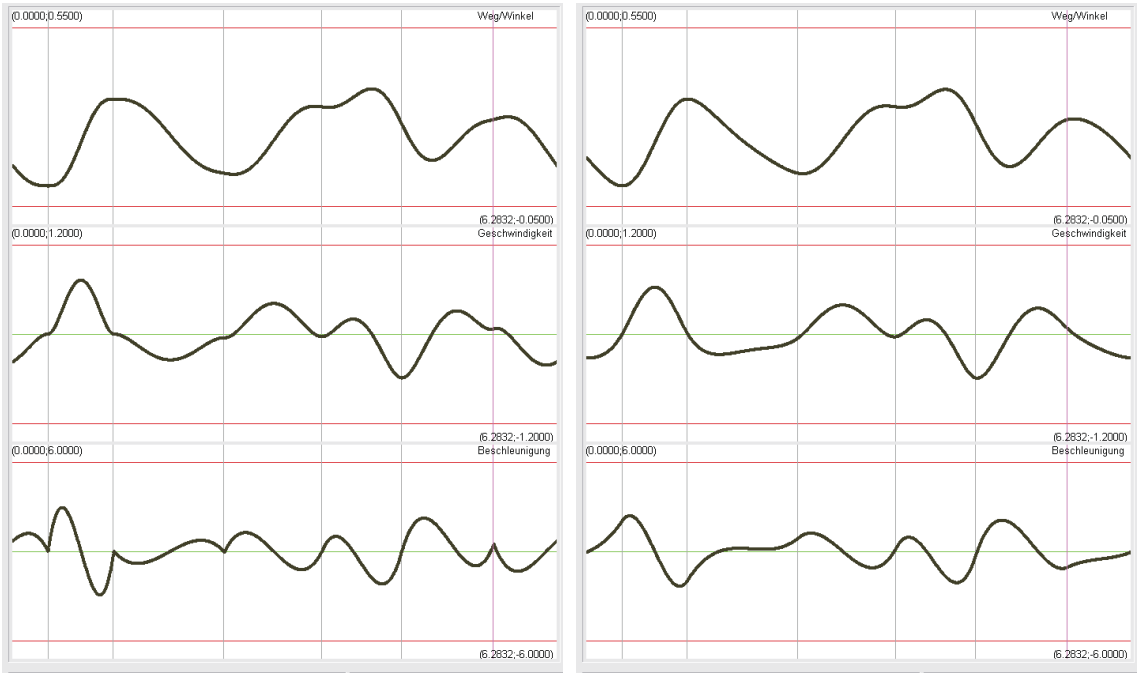


Abbildung 2.7: *Automatische Glättung der zweiten Ableitung bei Polynomen 5. Grades. Links: Vorher. Rechts: Nachher.*

jedoch sichergestellt, dass die Werte für x, y und v an jeder definierten Abschnittsgrenze erhalten bleiben.

2.1.6 Toleranzen

Braune [3] betont, dass die Variation von Stützpunkten im Rahmen zulässiger Toleranzen oft bessere Optimierungen liefert, als die Suche nach einem idealen Bewegungsgesetz. Um diesem Aspekt nachzugehen, kann das Konzept globaler Minimal- und Ma-

x_{min}	x	x_{max}
y_{min}	y	y_{max}
v_{min}	v	v_{max}
a_{min}	a	a_{max}

Tabelle 2.2: *Toleranzen für Parameter an Abschnittsgrenzen*

ximalwerte kann weiter ausgebaut werden, indem zu jedem Parameter an jeder Grenze ein Minimal- und ein Maximalwert vorgegeben wird. Dadurch steigt die Zahl der Parameterwerte einer Abschnittsgrenze von vier auf zwölf (s. Tab. 2.2). Der Mehraufwand für den Nutzer durch zusätzlich notwendige Eingaben kann in Grenzen gehalten werden, indem die Toleranzen als Differenz zum zugehörigen Parameter gespeichert und mit einem Standardwert belegt werden können. Um Verläufe zu erzeugen, die beispielsweise einer Rast oder einer Gerade nahe kommen sollen, können zusätzlich für jeden Funktionsabschnitt Grenzen für die y-, v- und a-Werte zwischen benachbarten Abschnittsgrenzen festgelegt werden (s. Abb. 2.8). Da jetzt für Parameter Toleranzen exakt vorgegeben werden können, ist es auch möglich den Kurvenverlauf unter Berücksichtigung dieser Toleranzen automatisiert zu optimieren. Ziel dieser Optimierungen könnte beispielsweise Senkung der realen Maximalwerte für die Beschleunigung des Abtriebsglieds über den gesamten Funktionsverlauf sein. Auf die notwendigen Algorithmen zur Optimierung soll in dieser Arbeit jedoch nicht näher eingegangen werden.

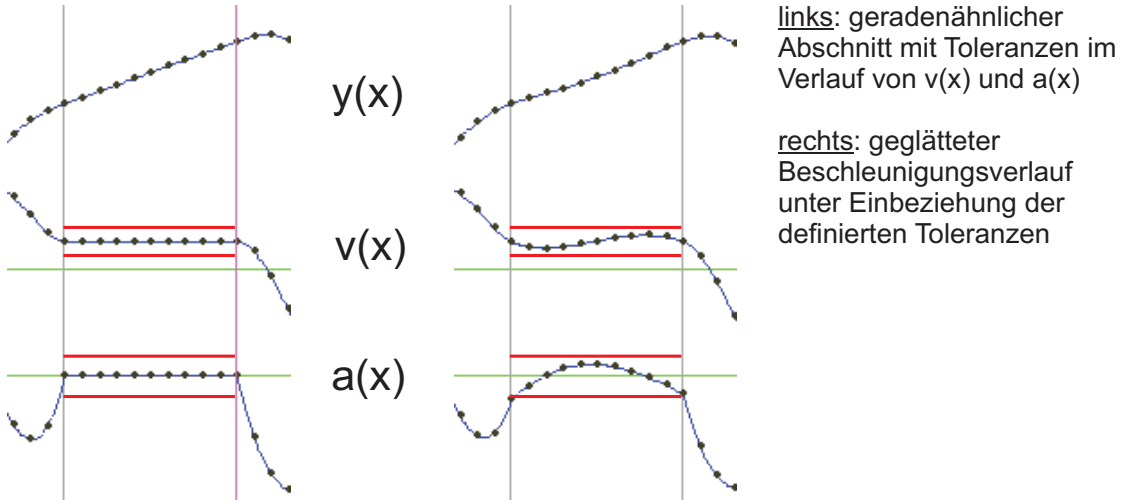


Abbildung 2.8: *Toleranzen für Parameter eines Abschnitts*

2.2 Getriebeauslegung

In diesem Kapitel soll eine Möglichkeit zur Beschreibung eines Kurvenscheibengetriebes vorgestellt werden. Dabei werden die vier bekanntesten Getriebetypen näher

beschrieben. Es handelt sich um Kurvenscheiben mit Abtriebsgliedern in Form von Rollenstößeln, Rollenhebeln, Flachstößeln und Flachhebeln, welche auch in den VDI-Richtlinien [18] erläutert wurden.
Generell wird bei der Parametrisierung oder Auslegung der Getriebe davon ausgegangen, dass sich das Rotationszentrum der Kurvenscheibe im Ursprung des kartesischen Koordinatensystems befindet.

2.2.1 Rollenstößel

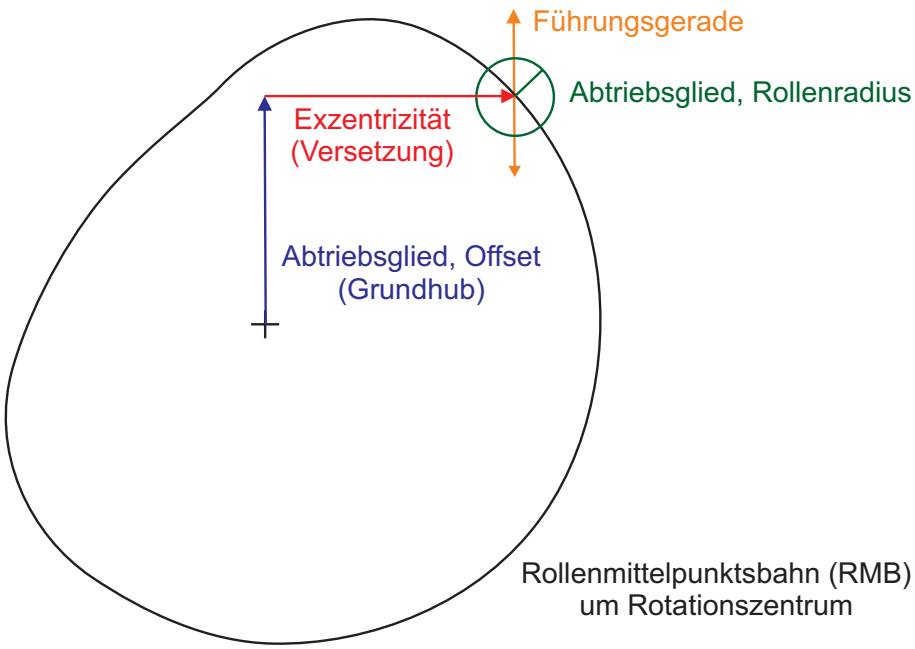


Abbildung 2.9: *Auslegung eines Getriebes mit Rollenstößel*

Ein Rollenstößel ist eine entlang einer Geraden geführte Rolle. Die Ausgangsposition des Mittelpunkts der Rolle wird durch die Koordinaten (Exzentrizität ; Offset) beschrieben (s. Abb. 2.9). Die Gerade, auf der die Rolle geführt wird, verläuft Parallel zur vertikalen Achse. Ist eine andere Richtung gewünscht, so ist dies möglich, indem das gesamte Getriebe gedreht wird. Durch die Übertragungsfunktion wird die Rollenmittelpunktsbahn (RMB) beschrieben. Die eigentliche Kontur der Kurvenscheibe findet sich im Abstand entsprechend des Rollenradius wieder (s. Abb. 2.10).

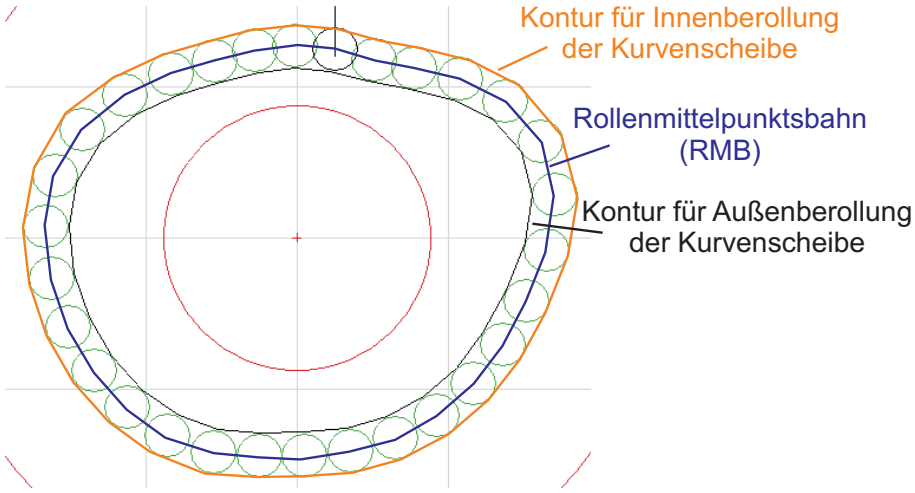


Abbildung 2.10: *Rollenmittelpunktsbahn und mögliche Kurvenscheibenkonturen*

2.2.2 Rollenhebel

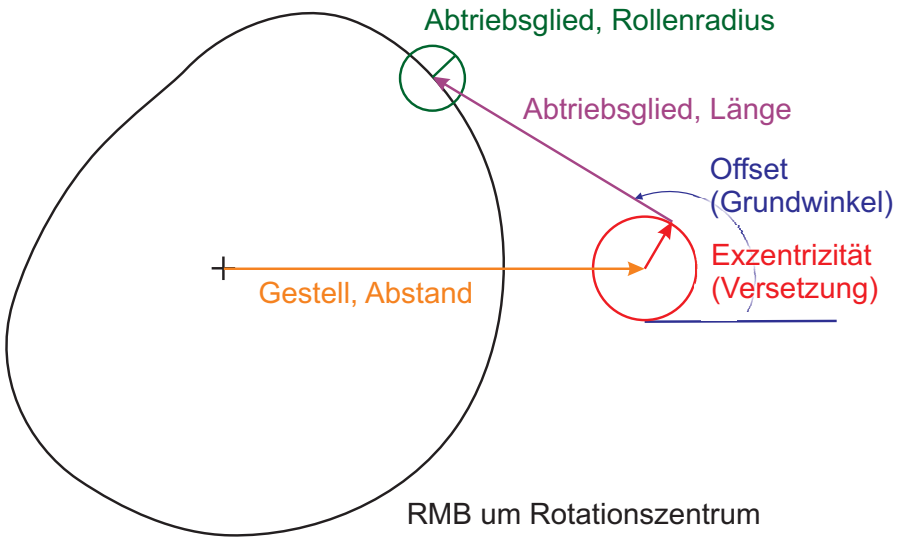


Abbildung 2.11: *Auslegung eines Getriebes mit Rollenhebel*

Ein Rollenhebel ist eine entlang einer Kreisbahn geführten Rolle. Die Parametrisierung wird in Abb. 2.11 veranschaulicht. An den Koordinaten (Gestellabstand ; 0) findet sich das Rotationszentrum des Abtriebsgliedes. Eine andere vertikale Position ist wieder durch Drehung des gesamten Getriebes möglich. Der Hebel befindet sich in einem gewissen Abstand zu diesem Zentrum, der Exzentrizität. Die Exzentrizität beschreibt

demnach den Radius eines Kreises, um den sich das eine Ende des Hebels dreht. Von diesem Punkt ist das Zentrum des Abtriebsgliedes eine definierte Länge entfernt. Der Offset beschreibt den Ausgangswinkel des Hebels.
Vor allem beim Rollenhebel, aber auch beim Rollenstößel, gilt es zu beachten, dass sich die Aufhängung der Rolle nicht in der selben Ebene befinden muss, wie die Abtriebsrolle bzw. die Kurvenscheibe. Somit sind Schnitte des Hebels mit der Kurvenscheibe in der zweidimenstionalen Ansicht im Allgemeinen als zulässig anzusehen.

2.2.3 Flachstößel

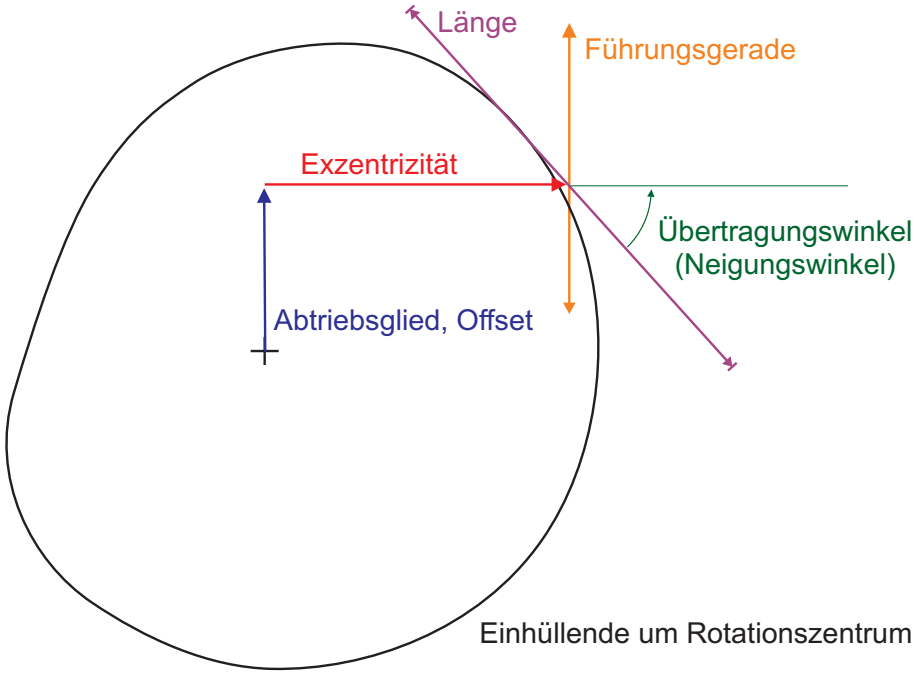


Abbildung 2.12: *Auslegung eines Getriebes mit Flachstößel*

Ein Flachstößel ist geometrisch gesehen eine Strecke, die entlang einer Geraden geführt wird. Die Ausgangsposition des Mittelpunktes eines Flachstößels wird wie beim Rollenstößel beschrieben (s. Abb. 2.12). Wie auch beim Rollenstößel liegt diese Gerade parallel zur vertikalen Achse. Die Strecke hat neben einer definierten Länge auch einen definierten Winkel. Ist dieser Winkel 0, so liegen Strecke und Gerade im rechten Winkel zueinander. Der Berührpunkt mit der Kurvenscheibe liegt irgendwo auf der

Strecke. Über die Übertragungsfunktion wird eine Schar dieser Strecken definiert, die die Kurvenscheibe einhüllen.

2.2.4 Flachhebel

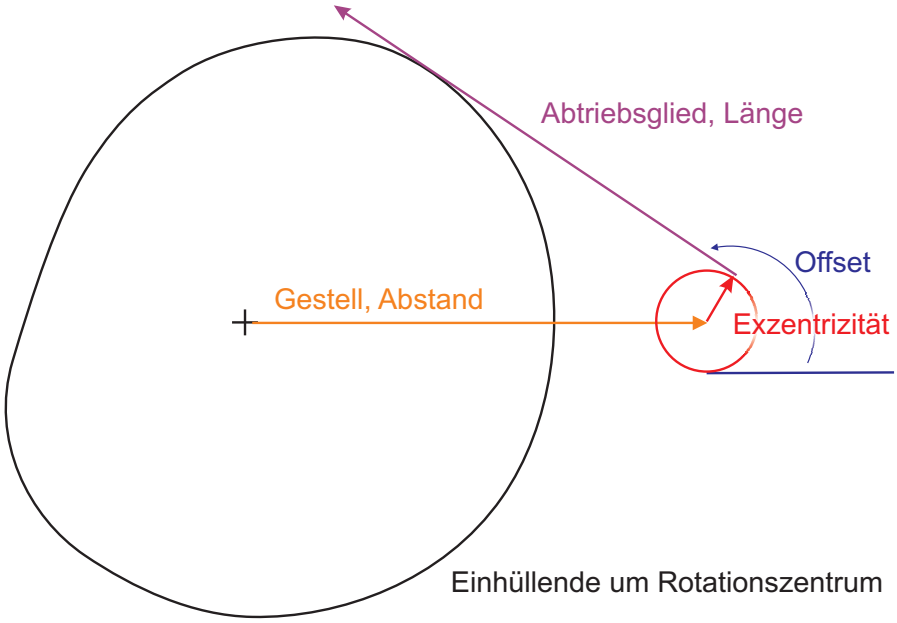


Abbildung 2.13: *Auslegung eines Getriebes mit Flachhebel*

Ein Flachhebel (s. Abb. 2.13) rotiert (ggf. mit einer definierten Exzentrizität) um das eine Ende. Die einhüllende Geradenschar, die durch die Übertragungsfunktion und die Parametrisierung des Getriebes definiert wird, erzeugt die Kurvenscheibe.

2.3 Synthese der Kurvenscheibe

Ziel der Kurvenscheibensynthese ist es, das Profil der Kurvenscheibe zu ermitteln. Sowohl die Vorgehensweise als auch die Rechenmethoden können sich hierbei teilweise stark unterscheiden. In der VDI 2142 [18, S. 21] wird folgende Unterscheidung vorgenommen:

1. Geometrie

2. analytische Geometrie

3. komplexe Zahlen

In dieser Arbeit wird vor allem Wert auf einfaches Verständnis und Nachvollziehbarkeit der Rechnungen gelegt, um daraus entstehende Software kosteneffizient warten zu können. Aus diesem Grund werden möglichst einfache und in Zeichnungen schnell nachvollziehbare geometrische Zusammenhänge in die Rechnung integriert.

2.3.1 Virtuelle Pressung

In einem Hauptseminar [20] wurde ein alternatives Verfahren zur Profilberechnung entwickelt. Hierbei wird das Profil von durch das Abtriebsglied begrenzten Mittelpunktstrahlen, deren Ursprung im Rotationszentrum der Kurvenscheibe liegt, berechnet. Zunächst haben alle Mittelpunktstrahlen eine unendliche Länge. Das Abtriebsglied begrenzt alle Strahlen, die es schneidet. Wird die Scheibe weitergedreht und die Position des Abtriebsglieds dementsprechend geändert, werden bisher nicht begrenzte Strahlen ggf. begrenzt und ein Teil der bisher begrenzten Strahlen eventuell weiter begrenzt. Das Ergebnis sind Stützpunkte des berechneten Profils. Diese Methode weist zwei Nachteile auf. Zum einen müssen bei einer hohen Auflösung sehr viele Berechnungen pro Winkelstellung des Antriebsgliedes durchgeführt werden. Zum anderen wird die Validierung der entstehenden Kurvenscheibe erschwert, da nicht sichergestellt ist, ob das Abtriebsglied in jeder Winkelstellung Kontakt zur Kurvenscheibe hat oder die Strahlen durch die benachbarten Abtriebsgliedpositionen zu stark begrenzt werden. Dieses Manko könnte man umgehen, indem für jeden Mittelpunktstrahl die Winkelstellung mitgeführt wird, die ihn am stärksten begrenzt. Das Kurvenprofil kann nur dann gültig sein, wenn jede Winkelstellung am Ende wenigstens einen Mittelpunktstrahl begrenzt. Nimmt man die Anzahl der Mittelpunktstrahlen mit m an und die Anzahl der in der Rechnung berücksichtigten Winkelstellungen der Kurvenscheibe mit n, so müssen im schlimmsten Fall für jeden Winkel alle Mittelpunktstrahlen auf erneute Begrenzung geprüft werden. Aufgrund der dadurch entstehenden Komplexität von $O(m \cdot n)$ fand dieses Verfahren zugunsten der direkten Berührpunktermittlung im weiteren Verlauf der Arbeit keine Berücksichtigung.

2.3.2 Profil durch Berührpunktermittlung

Die Alternative zur eben vorgestellten virtuellen Pressung besteht darin, zu jeder Winkelstellung der Kurvenscheibe den Berührpunkt des Abtriebsglieds mit dieser direkt zu berechnen. Dadurch wird die Komplexität der Berechnung linear abhängig von der Anzahl der zu berechnenden Winkelstellungen. Außerdem ist dadurch ohne weiteres sichergestellt, dass das Abtriebsglied in jeder Winkelstellung auch einen Beitrag zur Kurvenscheibe liefert, was Voraussetzung für die Validität bzgl. der Übertragungsfunktion ist.

2.3.3 Abtastung und Interpolation

Mit der Abtastzahl wird die Anzahl der Winkelstellungen der Antriebskurvenscheibe bezeichnet, die in die Berechnung für das Profil der Kurvenscheibe einfließen. Je feiner die Übertragungsfunktion definiert ist, desto höher muss diese Abtastzahl für eine hinreichende Nachbildung sein. Aber nicht nur die Definition der Übertragungsfunktion hat Einfluss auf die Anzahl der benötigten Abstastungen. Auch die Entstehung der Kurvenscheibe aus den berechneten Stützpunkten spielt eine nicht unerhebliche Rolle. Zur Vereinfachung wird davon ausgegangen, dass der Winkel zwischen benachbarten Abtastungen äquidistant ist. Optimierungen sind zwar möglich, wenn in bestimmten Bereichen mehr Abstastungen durchgeführt werden, dies gilt jedoch auch, wenn man die Abtastzahl schlicht allgemein erhöht. Zwischen den ermittelten Stützpunkten kann die Kontur der Kurvenscheibe durch Interpolation geschätzt werden.

Kartesiche Koordinaten

Im einfachsten Fall bilden die ermittelten Punkte die Eckpunkte eines Polygons im kartesischen Koordinatensystem (s. Abb. 2.14). Selbst wenn die Übertragungsfunktion beispielsweise eine Rast beschreibt, so muss die Abstastzahl dennoch sehr hoch sein, damit der daraus resultierende Kreisbogen auf der Kurvenscheibe nachgebildet wird. Dennoch hat diese Beschreibung der Kurvenscheibe vor allem in der Computergrafik ihre Berechtigung, da hier Darstellungen und Berechnungen zumeist auf Polygonen bzw. Dreiecken basieren.

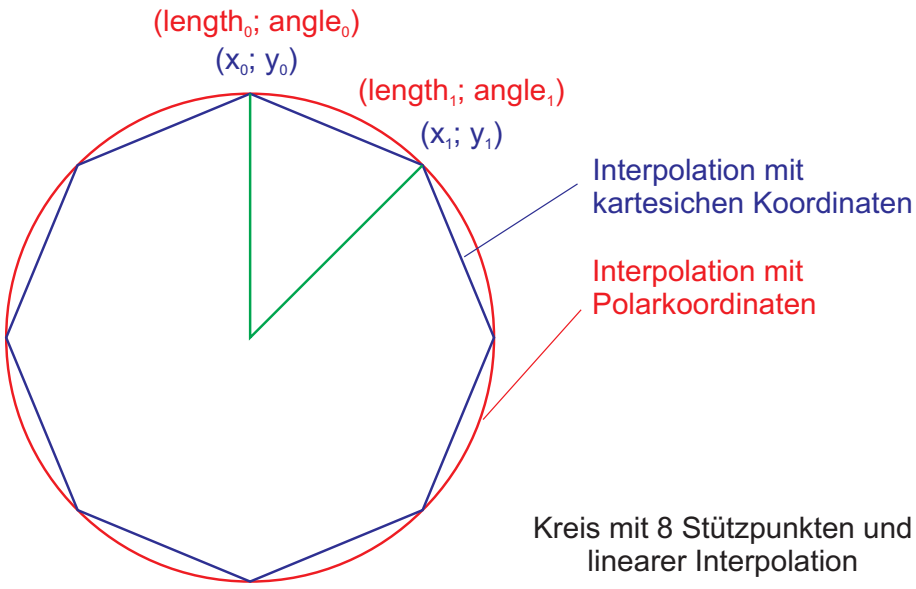


Abbildung 2.14: *Lineare Interpolation anhand von 8 Stützpunkten.*

Polarkoordinaten

Eine bessere Variante beschreibt die Stützpunkte in Polarkoordinaten mit Winkel und Radius. Zwischen den benachbarten Winkeln muss nun der Radius interpoliert werden (s. Abb. 2.14). Selbst bei linearer Interpolation ist das Ergebnis bereits besser als bei Verwendung eines Polygons mit kartesischen Koordinaten. Eine Rast kann hier mit den die Rast begrenzenden Stützpunkten perfekt beschrieben werden. Knicke an den Stützpunkten können jedoch nur mit besseren Interpolationsmethoden ausgeschlossen werden.

2.3.4 Berechnung der Rollenmittelpunktsbahn (RMB)

Die Rollenmittelpunktsbahn muss bei allen Getrieben mit einer Rolle als Eingriffsglied ermittelt werden. Beispiele hierfür sind der Rollenstößel (s. Abschnitt 2.2.1) und der Rollenhebel (s. Abschnitt 2.2.2). Die Position des Eingriffsgliedes bestimmt sich als Addition des Funktionswertes zum Offset der Getriebeparameter (s. Abb .2.15). Die jeweilige Lage der Abtriebsrolle ist also über einfache geometrische Berechnungen möglich. Da diese Lage (im Bild grün dargestellt) zu einem bestimmten Winkel der

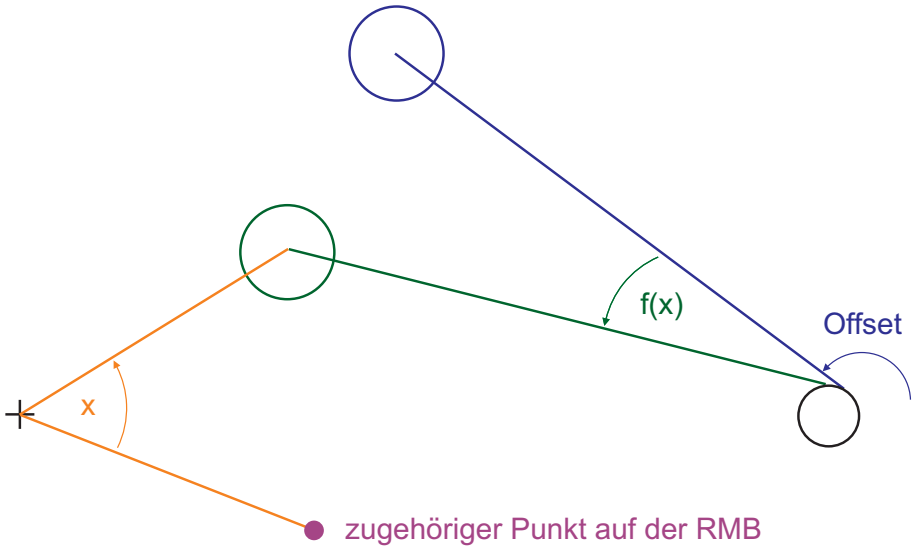


Abbildung 2.15: *Berechnung der RMB*

Kurvenscheibe gehört, muss dieser Punkt noch entgegen der Drehrichtung der Kurvenscheibe (im Bild dreht die Scheibe gegen den Uhrzeigersinn) um das Rotationszentrum dieser gedreht werden. Die Summe aller so ermittelten Punkte bildet die Rollenmittelpunktsbahn oder kurz RMB.

2.3.5 Kurvenkontur aus RMB

Da die Abtriebsrolle einen festen Radius hat, liegt die Kurvenkontur äquidistant zur RMB. Es wird nun eine leicht nachvollziehbare und einfach zu implementierende Variante vorgestellt, wie die Kontur ermittelt werden kann. In Abb. 2.16 repräsentieren die Kreise die Abtriebsrolle um die ermittelten Punkte der RMB. Die eingezeichneten Durchmesser dieser Kreise stellen die Normale dar. Die Schnittpunkte der Kreise mit ihrer Normalen sind die Berührpunkte. Hier ist auch schön zu sehen, dass mit dieser Berechnung sowohl eine Außenkurve als auch eine Innenkurve berechnet werden kann. Die Normale lässt sich ermitteln, indem man den infinitesimalen Nachbarpunkt auf der RMB betrachtet. Verbindet man diese Punkte, so erhält man die Tangete. Im rechten Winkel dazu liegt die Normale. Eine einfache numerische Annäherung des infinitesimalen Nachbarpunkts kann erreicht werden, indem der Punkt auf der RMB bestimmt

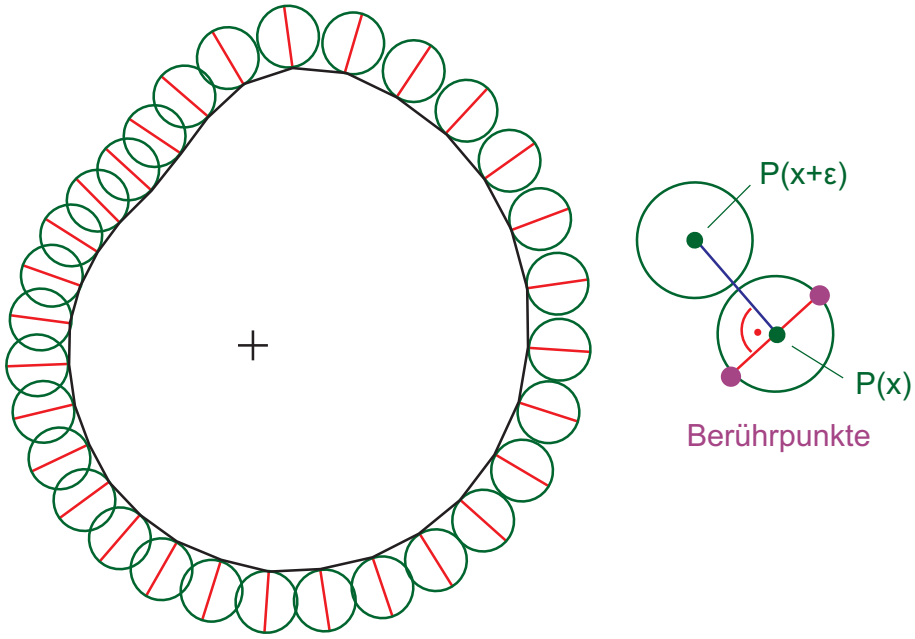


Abbildung 2.16: *Konturermittlung aus der RMB*

wird, der beispielsweise $\epsilon = 10^{-8}$ (im Bogenmaß der Kurvenscheibe) vom ermittelten RMB-Punkt entfernt ist. Dadurch werden praktisch zwar doppelt so viele Abtastungen vorgenommen, wie Stützpunkte der Kurvenscheibe gewünscht sind, jedoch sind die Berechnungen sehr einfach, sodass hinsichtlich der Geschwindigkeit keine Nachteile gegenüber einer analytischen Methode entstehen. Aus den beiden Punkten lässt sich über einfache Geometrie die Tangente und somit auch die Normale ermitteln. Vom ermittelten Punkt der RMB entlang der Normalen befinden sich die beiden Berührpunkte für die Innen- bzw. Außenkurve in einer Entfernung, die dem Radius der Abtriebsrolle entspricht.

2.3.6 Stützpunktermittlung bei flachem Eingriffsglied

Flache Eingriffsglieder, wie der Flachstößel (s. Abschnitt 2.2.3) oder der Flachhebel (s. Abschnitt 2.2.4), bilden eine Geradenschar $G(x)$, die die Kurvenscheibe einhüllt (s. Abb. 2.17). Wenn die Übertragungsfunktion gültig sein soll, muss jede Gerade $g(x)$ einen Berührpunkt mit der Kurvenscheibe haben. Insbesondere muss dieser Berührpunkt zwischen den Schnittpunkten mit den Nachbargeraden der Kurvenschar liegen.

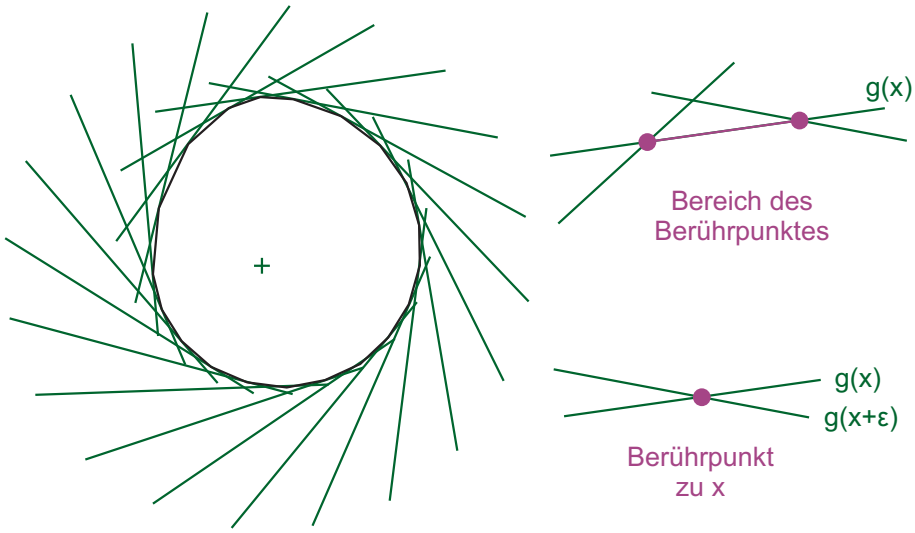


Abbildung 2.17: *Konturermittlung bei flachem Eingriffsglied*

Wie auch schon bei der Ermittlung der Kurvenkontur aus der RMB, gibt es auch hier eine einfache numerische Methode. Man ermittle die um beispielsweise $\epsilon = 10^{-8}$ (im Bogenmaß der Kurvenscheibe) entfernte Gerade $g(x + \epsilon)$. Der Schnittpunkt von $g(x)$ mit $g(x + \epsilon)$ ist dann der Berührpunkt zu $g(x)$.
Die Begründung hierfür ist, dass das Abtriebsglied in jeder Winkelstellung einen Berührpunkt mit der Kurvenscheibe haben muss, damit die Übertragungsfunktion erfüllt werden kann. Die benachbarten Geraden der Geradenschar schränken die mögliche Position des Berührpunktes ein. Bei unendlicher Genauigkeit fällt der Schnittpunkt der Geraden mit beiden Nachbargeraden auf genau einen Punkt.
Zu möglichen numerischen Problemen siehe auch Abschnitt 4.1.5.

Kapitel 3

Interaktives Editieren

In diesem Kapitel werden einige Konzepte vorgestellt, die eine einfache Bearbeitung eines Kurvenscheibengetriebes innerhalb eines Programms mit grafischer Oberfläche ermöglichen. Im Wesentlichen kann zwischen der Bearbeitung der Übertragungsfunktion und der Getriebeparametrisierung unterschieden werden. In einem Prototyp wurden einige der im folgenden Text vorgestellten Ansätze implementiert.

3.1 Übertragungsfunktion

Wie in Abschnitt 2.1 beschrieben, setzt sich die Übertragungsfunktion aus Teilfunktionen zusammen. Neben der eigentlichen Funktion spielen auch Geschwindigkeit (1.Ableitung) und Beschleunigung (2. Ableitung) eine Rolle. In Abb. 3.1 sieht man eine Oberfläche, mit der man die Übertragungsfunktion bearbeiten kann. Den Hauptbereich nehmen drei untereinanderliegende Bereiche ein. Der oberste repräsentiert die Funktion. Darunter wird der Geschwindigkeitsverlauf dargestellt und darunter der Beschleunigungsverlauf. Auf der rechten Seite lassen sich sämtliche Parameter via Maus und Tastatur direkt eingeben. Die Funktion wird hier beschrieben mit $y = f(x)$, ihre 1. Ableitung mit $v = f'(x)$ und die 2. mit $a = f''(x)$. Zum Bearbeiten der Funktion wird zunächst ein Bearbeitungsmodus für die Interaktion mit der Maus festgelegt (select & operate). Es wird dabei zwischen den Modi Hinzufügen, Verschieben und Löschen von Abschnittsgrenzen unterschieden.

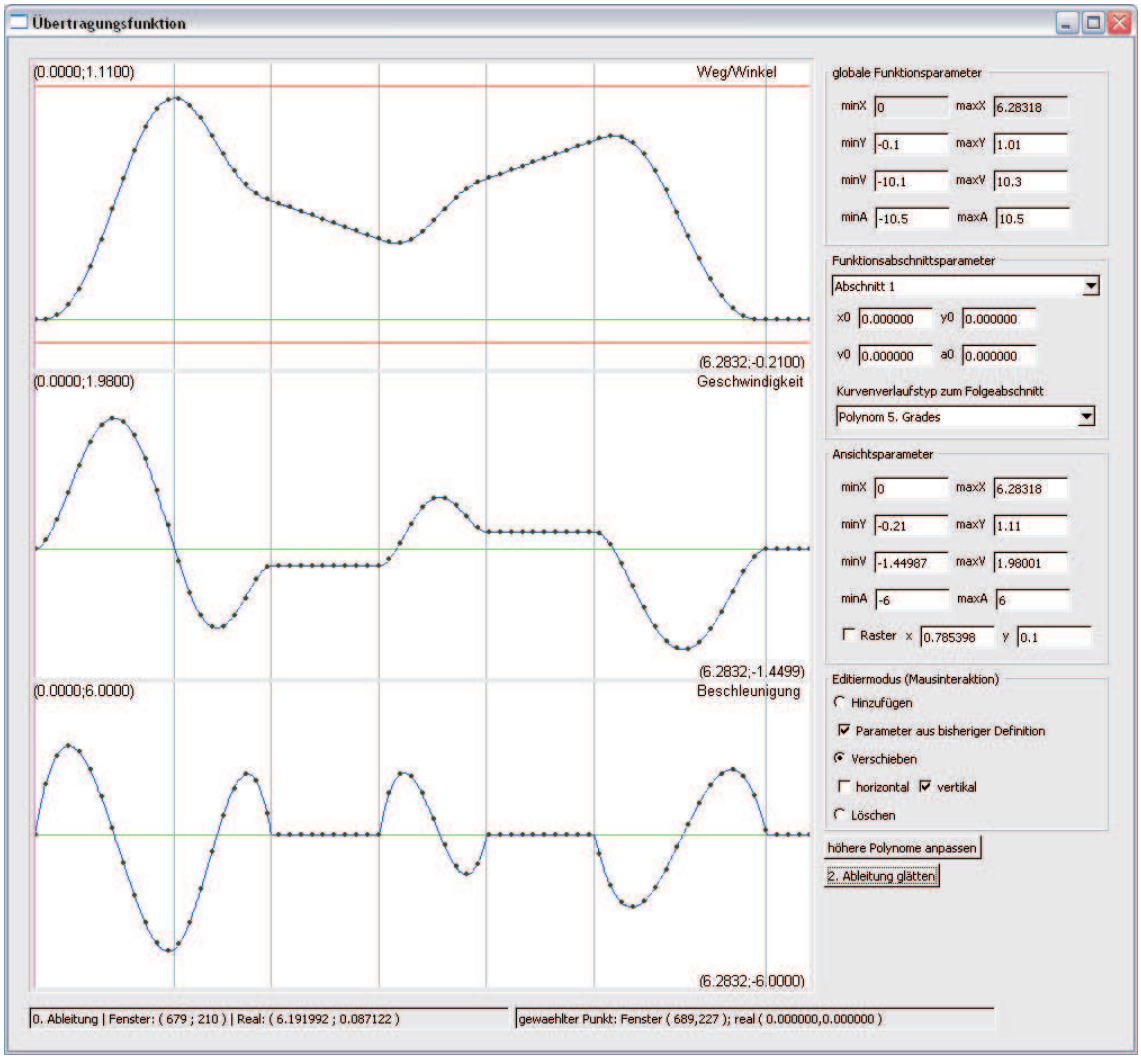


Abbildung 3.1: *Editor für die Übertragungsfunktion*

3.1.1 Hinzufügen

Ein Abschnitt kann hinzugefügt werden, indem man mit der Maus die gewünschte Position der linken Grenze des neuen Abschnitts in einem der drei Fenster anklickt. Diese Abschnittsgrenzen sind in Abb. 3.1 als vertikale Linien gekennzeichnet. Mit dem Klick in eines der Fenster sind zwei Parameter der neuen Abschnittsgrenze bestimmt. Das sind zum einen der x-Wert und zum anderen je nach Bereich der y, v oder a-Wert. Zur vollständigen Beschreibung einer Abschnittsgrenze fehlen also zumindest noch die Werte aus den beiden anderen Bereichen. Es gibt mehrere Möglichkeiten, diese Werte

zu ergänzen. Die einfachste ist die nicht gesetzten Werte einfach mit 0 zu belegen. Wird also beispielsweise eine Funktion ausschließlich im oberen Fenster definiert, würden an allen Abschnittsgrenzen die Geschwindigkeit und Beschleunigung des Abtriebsglieds mit 0 initialisiert. Vor dem Hinzufügen der neuen Abschnittsgrenze war ggf. bereits eine Funktion definiert. Demzufolge könnte man die fehlenden Werte auch von den alten Werten an der entsprechenden Position übernehmen. Zwischen diesen Modi könnte man beispielsweise über eine Umschalttaste oder Auswahlbuttons wechseln.
Sind alle Werte für die neue Abschnittsgrenze definiert, gilt es den Verlaufstyp festzulegen. Links der Grenze gilt der Verlaufstyp, der mit der linken Nachbargrenze definiert wurde. Rechts davon und bis zur rechten Nachbargrenze gilt der eigene Typ. Auch hier sind zwei Optionen denkbar. Zum einen kann der eigene Typ mit einem Standardtyp (vorzugsweise Polynom 5. Grades) initialisiert werden oder aber den vorher an der Stelle gültigen Typ übernehmen. Letzteres ist jedoch weniger empfehlenswert, da im Falle einer Geraden nun zwei Geraden benachbart sind, womit keine Stoßfreiheit gewährleistet werden kann.
Zur Berechnung des Verlaufs innerhalb eines Abschnitts sind stets die Parameter beider Grenzen nötig. Beim Hinzufügen eines Abschnitts müssen also immer zwei Verläufe neu berechnet werden.

3.1.2 Verschieben

Ist der Modus Verschieben für die Maus aktiv, kann man die Parameter der Abschnittsgrenzen in den drei Bereichen ändern. Hier gilt es zunächst auszuwählen, welche Grenze behandelt werden soll. Eine einfache Methode ist es, mit einem einfachen Klick die neue Position zu bestimmen. Die Abschnittsgrenze, die näher an diesem Punkt liegt, wird an diese Position verschoben. Hier ist es sinnvoll durch entsprechende Schalter auszuwählen, ob sowohl x- als auch y- bzw. v- oder a-Werte verschoben werden sollen, oder nur eine Komponente. Um Verwechslungen zu vermeiden, kann hier die Gestalt des Mauszeigers auf den aktuell gewählten Modus hinweisen. In jedem Fall sollten die Werte in den beiden anderen Bereichen nicht automatisch verändert werden, da dies schnell zu Verwirrungen führt. Ein nicht unwesentlicher Nachteil dieser Methode ist,

dass eine Grenze maximal um die Hälfte der Distanz zur Nachbargrenze versetzt werden kann, da ansonsten die Nachbargrenze als die zu versetzende erkannt wird. Dieses Problem ließe sich umgehen, indem man eine Zweiklick-Methode implementiert. Der erste Klick wählt die zu versetzende Grenze, der zweite bestimmt die Zielposition. Ein ähnliches Verhalten ist auch mit einer Drag-Funktionalität realisierbar.
Eine zusätzliche sinnvolle Funktion wäre es, einen kompletten Abschnitt zu verschieben, sprich die beiden anliegenden Grenzen gleichermaßen zu versetzen.
Beim Verschieben einer Grenze müssen die beiden angrenzenden Funktionsverläufe neu berechnet werden. Fraglich ist, ob es sinnvoll ist zuzulassen, dass eine Grenze über andere Grenzen hinweg versetzt wird. Liegt der zulässige Bereich einer Grenze zwischen ihren Nachbargrenzen, erspart dies einige Sonderfälle und Verwirrungen.

3.1.3 Löschen

Das Entfernen eines Abschnitts ist recht simpel. Ein einfacher Klick in die Nähe der linken Grenze des zu entfernenden Abschnitts genügt. Die Nachbargrenzen enthalten alle notwendigen Informationen zur Neuberechnung des durch die gelöschte Grenze geteilten Abschnitts.

3.1.4 Ansicht

Besonders wichtig bei der Darstellung der Übertragungsfunktion ist, dass die verschiedenen Ableitungen synchron betrachtet werden können, d.h. dass die drei Fenster stets dieselbe Breite haben sollten und dabei denselben Bereich von x-Werten anzeigen. Insbesondere bei Zoomfunktionen ist das von essentieller Bedeutung. Die einfachste Möglichkeit dies sicherzustellen ist es, lediglich eine Skalierung der vertikalen Achse zuzulassen und stets den vollen Bereich der gültigen x-Werte anzuzeigen. Ein Monitor mit einer horizontalen Auflösung von mindestens 1600 Pixeln ist dabei als sinnvoll zu erachten, um die Details hinreichend darstellen zu können.
Bei der Bearbeitung der Übertragungsfunktion kommt es nicht selten vor, dass eine Ableitung über den angezeigten vertikalen Bereich hinausragt oder aber sehr flach verläuft. Dann ist es sinnvoll, den vertikalen Bereich anzupassen. Das Scrollrad der

Maus ist hier für eine Skalierung (Zoom) prädestiniert. Welcher Bereich dabei als Zentrum des Zooms dient, kann durch die Position des Mauszeigers bestimmt werden. Zusätzlich kann es sinnvoll sein, mit der rechten Maustaste das gewünschte Zentrum der Ansicht auszuwählen, um eine Verschiebung zu realisieren.
In einer Statusleiste können die zur Mauszeigerposition gehörigen Koordinaten ausgegeben werden. Zur leichteren Orientierung sollte die 0 auf der vertikalen Achse hervorgehoben sein und ggf. ein Raster angezeigt werden. Auch ist es sinnvoll, den zulässigen Wertebereich zu kennzeichnen. Im einfachsten Fall sind dies wie in Abb. 3.1 rot dargestellte globale Minimal- und Maximalwerte für die Funktion und deren Ableitungen, die natürlich vom Nutzer verändert werden können.

3.1.5 Temporäre Definition

Auswirkungen von beliebig vielen Änderungen

Um die Auswirkungen von Änderungen an der Funktionsdefinition zu visualisieren, empfiehlt es sich, mit zwei Funktionsdefinitionen zu arbeiten. Sobald eine Definition einen Stand erreicht hat, den man behalten möchte, bestätigt man die Funktion mit einem Übernehmen-Button. Diese bestätigte Funktion wird eingezeichnet, ebenso ihre Ableitungen. Gleichzeitig kann an der temporären Funktionsdefinition weitergearbeitet werden. Somit können die Unterschiede beider Definitionen nach beliebig vielen Bearbeitungsschritten (Hinzufügen, Löschen oder Verändern von Abschnitten) visualisiert werden. Über einen Verwerfen-Button kann die temporäre Definition durch eine Kopie der bereits bestätigten ersetzt werden. Zur Berechnung der Kurvenscheibe dient dann die zuletzt bestätigte Definition als Grundlage.

Auswirkungen einer Änderung

Für die Visualisierung der Auswirkungen einer Veränderung ist eine Drag-Funktionalität besonders geeignet. Der Drag-Modus wird aktiv, wenn die linke Maustaste im Fenster gedrückt wird. Er wird beendet, wenn die linke Maustaste wieder losgelassen wird oder die Aktion mit der rechten Maustaste abgebrochen wird. Während der Drag-Modus aktiv ist, wird neben der bisherigen Funktionsdefinition auch

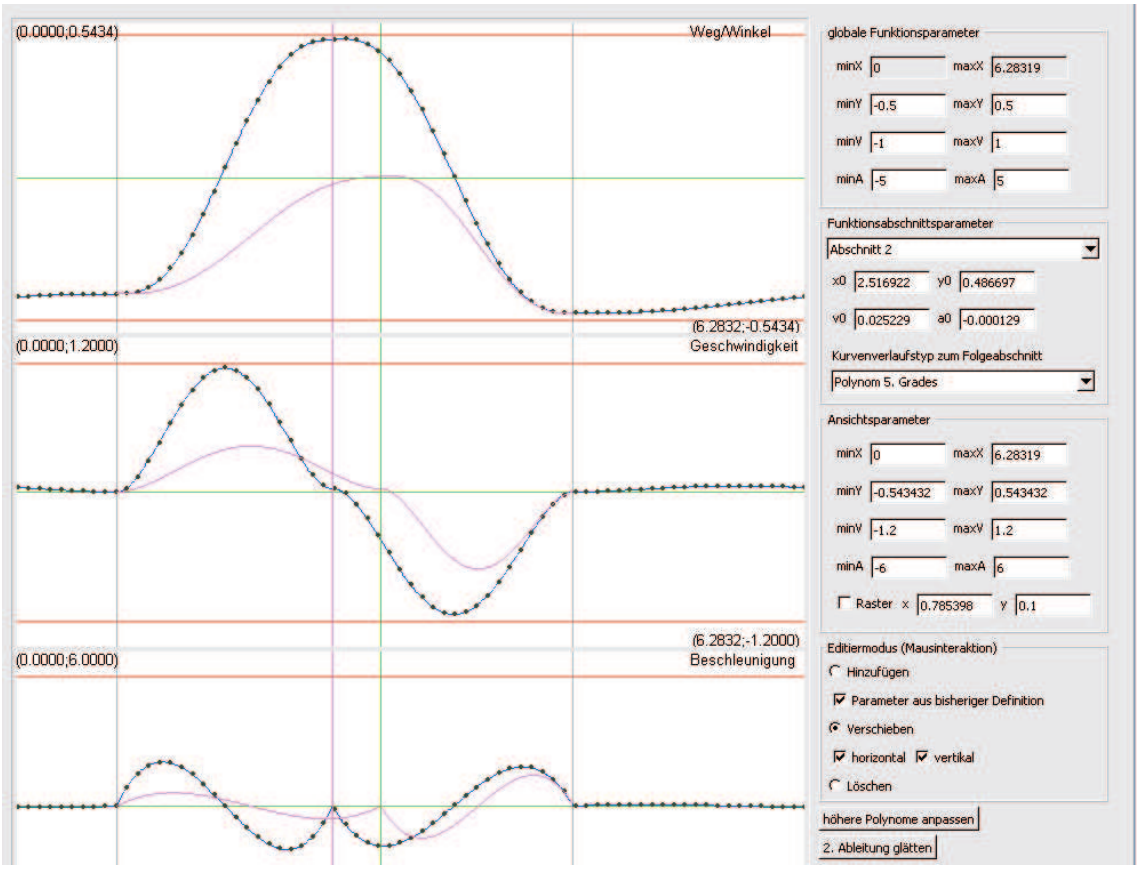


Abbildung 3.2: *Drag-Modus beim Verschieben einer Abschnittsgrenze. Die magentafarbene Kurve zeigt den Kurvenverlauf, wenn das Verschieben der mittleren Abschnittsgrenze bestätigt wird.*

der Kurvenverlauf dargestellt, wie er durch Loslassen der linken Maustaste bestätigt werden würde. Ein Beispiel ist in Abb. 3.2 zu sehen.

3.2 Getriebeparametrisierung

Die Definition der Übertragungsfunktion und die Getriebeauslegung können zwar sequentiell ausgeführt werden, jedoch ist oftmals auch eine parallele Bearbeitung beider notwendig. Um eine hinreichende Detaildarstellung dennoch während der Bearbeitung zu gewährleisten, kann es sinnvoll sein, beide Editoren in Karteireitern oder verschiedenen Fenstern darzustellen. Letzteres hat den Vorteil, dass sowohl eine einzelne Betrach-

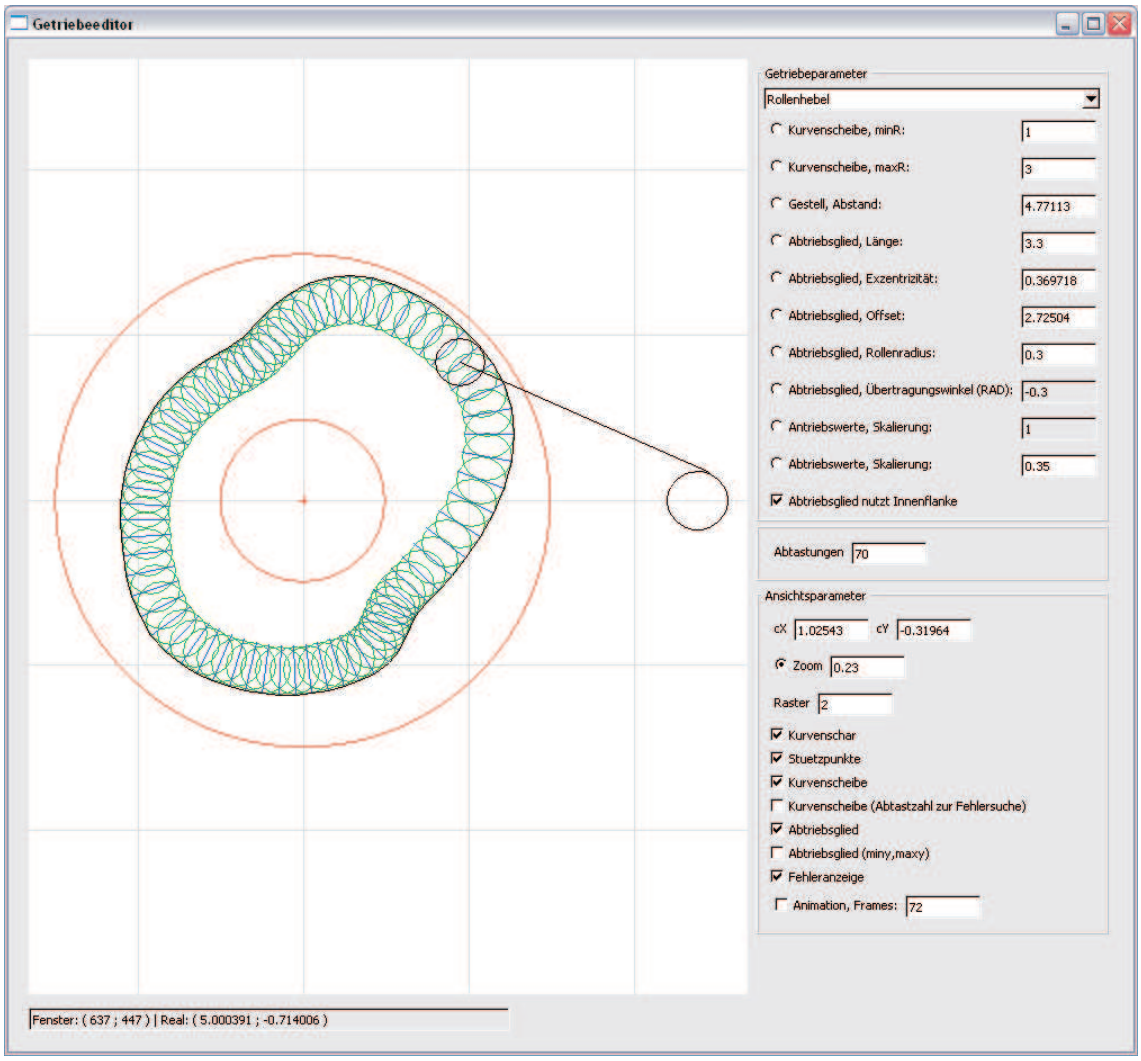


Abbildung 3.3: *Editor für die Getriebeauslegung*

tung wie auch eine synchrone Betrachtung der Editoren ermöglicht wird. Insbesondere bei der Nutzung mehrerer Monitore erleichtert dies die Erkennung der Zusammenhänge zwischen der Übertragungsfunktion und der Kontur der Kurvenscheibe. Ein solcher Editor für die Getriebeauslegung ist in Abb. 3.3 dargestellt.

In diesem Editor werden der Getriebetyp und dessen Parameter festgelegt und visualisiert. Außerdem kann zusammen mit der definierten Übertragungsfunktion die entstehende Kurvenscheibe angezeigt werden sowie die Kurvenscharen, die zu dieser geführt haben. Um die Auswirkungen der Änderung eines einzelnen Parameters zu ver-

deutlichen, kann über Auswahlbuttons das Scrollrad der Maus mit einem beliebigen Parameter verknüpft werden.

3.2.1 Zoom

Während der Tests mit dem Prototypen hat es sich als effizient erwiesen, mit dem Scrollrad den Zoomfaktor festzulegen und mit der rechten Maustaste das Zentrum der Ansicht zu bestimmen. Natürlich sollten auch Tastatureingaben für genauere Angaben möglich sein.

3.2.2 Zulässige Grenzen

Eine bessere Orientierung bietet die Möglichkeit, einen minimalen und einen maximalen Radius für die entstehende Kurvenscheibe anzugeben und anzuzeigen. Diese Radien sind unabhängig von den Berechnungen von Abtriebsglied und Kurvenscheibenkontur. Auf diese Weise wird schnell ersichtlich, ob bei Veränderung eines Getriebeparameters oder auch der Übertragungsfunktion diese zugelassenen Grenzen überschritten werden. Im Editor der Übertragungsfunktion konnten zulässige globale Minimal- und Maximalwerte definiert werden. Wird das Abtriebsglied in diesen Positionen eingezeichnet, wird deutlich, in welchem Bereich es sich nach diesen Festlegungen bewegen kann (s. Abb. 3.4), sofern die Übertragungsfunktion diese Grenzwerte nicht überschreitet.

3.2.3 Zuschaltbare Elemente

Zur Verbesserung der Übersichtlichkeit sollten sämtliche Elemente der Anzeige einzeln zu- und abschaltbar sein. Insbesondere sind folgende von Bedeutung:

- Polygon der Kurvenscheibe
- Abtriebsglied bei Antriebswinkel 0
- Kurvenschar des Eingriffsglieds
- Stützpunkte der Kurvenscheibe auf den Kurvenscharen

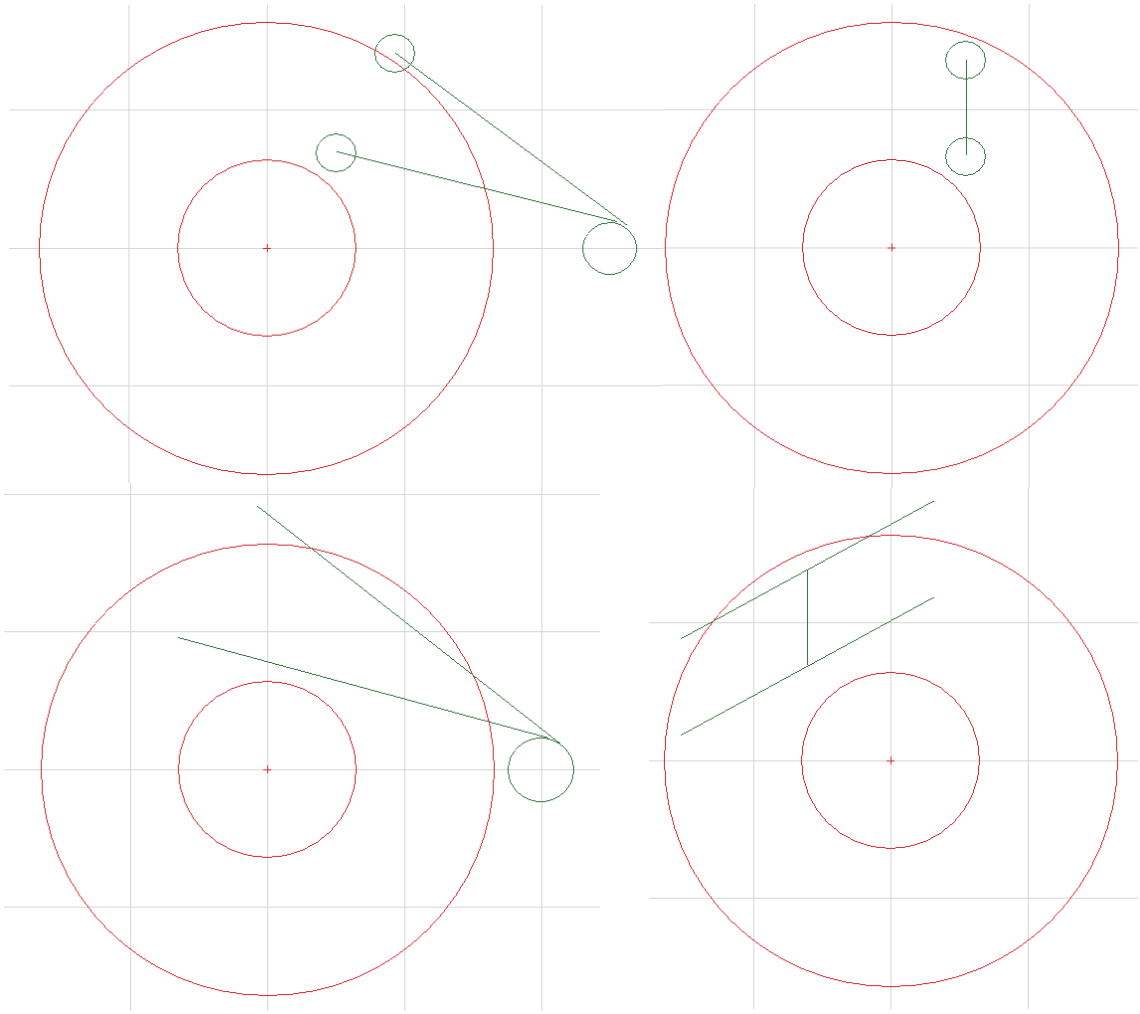


Abbildung 3.4: *Auslegung der zulässigen Grenzen*

- Abtriebsglied bei zulässigen Minimal- und Maximalwerten der Übertragungsfunktion
- zulässige Minimal- und Maximalradien der Kurvenscheibe
- Kurvenscheibe nach temporärer Definition

3.2.4 Wertskalierung

Im einfachsten Fall ist der x-Wert der Übertragungsfunktion als Winkel der Kurvenscheibe zu interpretieren. Man könnte jedoch auch eine Zeitspanne für eine Umdrehung

als Grundlage nehmen. Für diesen Fall kann der x-Wert mit einem Faktor versehen werden, um die Zeitspanne auf eine volle Umdrehung im Bogenmaß zu beziehen. Zusätzlich ist es mit diesem Faktor möglich den Drehsinn der Kurvenscheibe zu ändern, indem er ein negatives Vorzeichen erhält.
Auch die y-Werte der Übertragungsfunktion sowie der Offset des Abtriebsgliedes können mit einem Faktor versehen werden. Dadurch lässt sich beispielsweise der Hub eines Stößels schnell anpassen, ohne den prinzipiellen Verlauf der Übertragungsfunktion zu verändern.

3.3 Validierung

Über den Editor werden Stützpunkte für eine Kurvenscheibe generiert. Es stellt sich natürlich die Frage, ob diese Kurvenscheibe die definierte Übertragungsfunktion auch realisieren kann. Einige Probleme kann der Nutzer sofort im Editor erkennen und entsprechend reagieren. In diesem Abschnitt werden einige typische Probleme gezeigt und Lösungen zur Behebung vorgeschlagen. Es wird davon ausgegangen, dass das Eingriffsglied stets an der Kurvenscheibe aufliegt, sei es durch Kraft- oder Formschluss. Physikalische Zusammenhänge wie Verschleiß oder das Abheben des Eingriffsgliedes aufgrund von Massenträgheit finden hier keine Berücksichtigung.

3.3.1 Inkonsistenz in Wertskalierung für Antriebswinkel

In Abschnitt 2.1.5 wurde gezeigt, wie man durch Korrektur von Parametern die Übertragungsfunktion glätten kann, um Sprünge und Knicke zu vermeiden. Eine andere Inkonsistenz kann auftreten, wenn der Definitionsbereich der Übertragungsfunktion nicht mit der Wertskalierung des Antriebswinkels harmoniert. In Abb. 3.5 links muss der Definitionsbereich vergrößert werden oder aber der Faktor für die Wertskalierung, um einen Vollkreis abzudecken. Auf der rechten Seite ist das Gegenteil der Fall. Der korrekte Skalierungsfaktor kann aus dem Definitionsbereich der Übertragungsfunktion automatisch ermittelt werden $(2 \cdot \pi/(x_{max} - x_{min}))$. Der Definitionsbereich hingegen sollte nicht automatisch verändert werden, da dadurch bereits definierte Funktionsab-

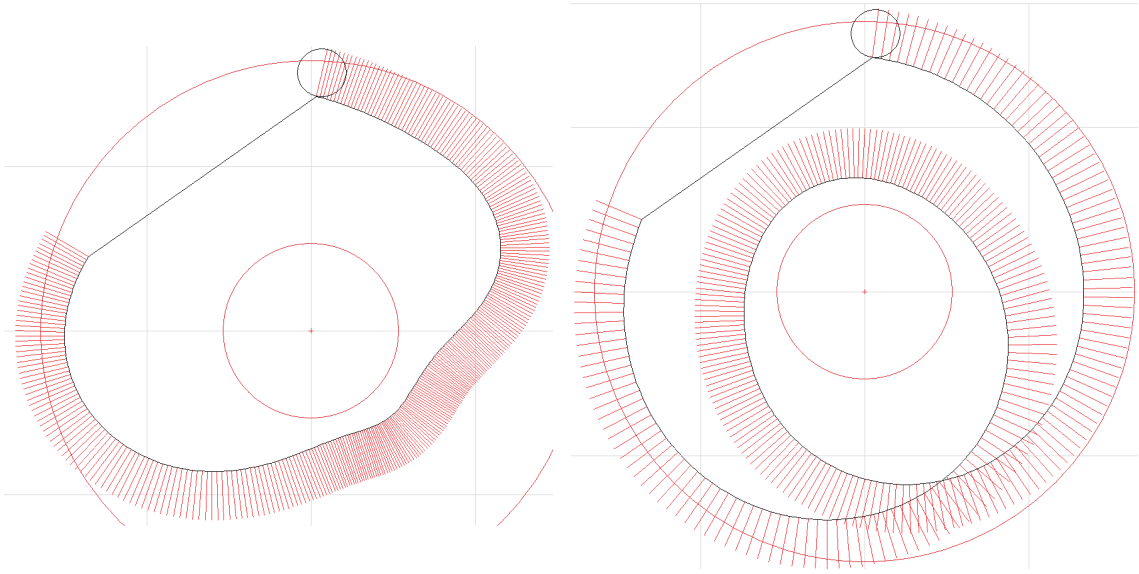


Abbildung 3.5: *Inkonsistenz bei Definitionsbereich bzw. Wertskalierung*

schnitte verändert werden würden.
Auf mögliche gewollte Skalierungen, die nicht zu einem Vollkreis führen, sondern eventuell für eine schwingend angetriebene Kurvenscheibe oder dreidimensionale Führungen genutzt werden können, soll in dieser Arbeit nicht näher eingegangen werden.

3.3.2 Starker Anstieg bei Abtriebsrolle

In Abb. 3.6 links ist eine Kurvenscheibe samt ihrer Stützpunkte zu sehen. Der Anstieg in der Übertragungsfunktion ist in der rot eingekreisten Zone zu steil. Rechts ist die Problemzone vergrößert dargestellt. Die Normalen der Abtriebsrolle schneiden sich, bevor der Berührpunkt mit der Kurvenscheibe erreicht ist. Für dieses Problem gibts mehrere einfache Lösungen:

- Korrektur der Übertragungsfunktion, Abflachen der Funktion
- Verwendung der anderen Flanke (sofern auf dieser nicht an anderer Stelle das gleiche Problem auftaucht)
- Erhöhung des Offsets, Die Kurvenscheibe erhält dadurch einen größeren Durchmesser

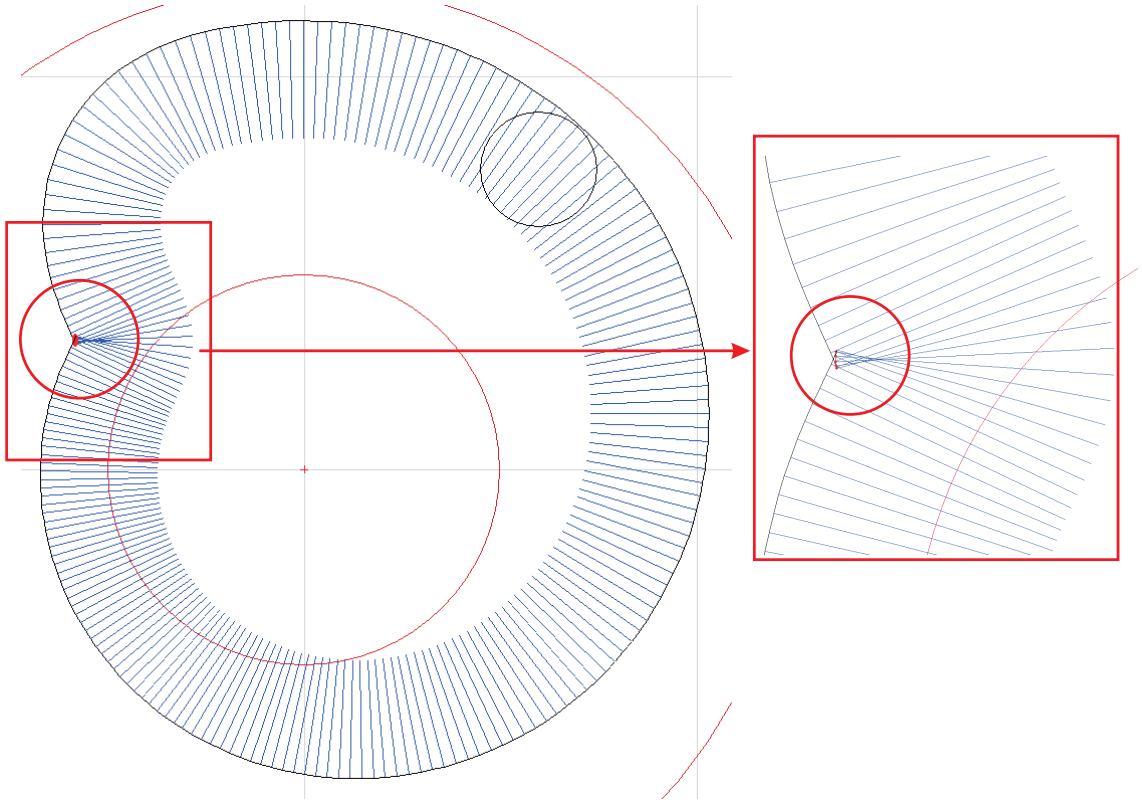


Abbildung 3.6: *Zu steiler Anstieg in der Übertragungsfunktion*

- Verkleinerung des Rollenradius

Streng monoton wachsende/fallende Winkel der Stützpunkte

Eine einfache Methode, diesen Fehler automatisch zu erkennen, besteht darin die Koordinaten der Stützpunkte in Polarkoordinaten zu betrachten. Den Koordinatenursprung bildet weiterhin das Rotationszentrum der Kurvenscheibe. Die Stützpunkte bilden eine geordnete Liste, die zur Beschreibung der Kurvenscheibe beispielsweise als Polygon dient. Innerhalb dieser Liste müssen die Winkel durchweg monoton steigen oder fallen, je nach Antriebsdrehrichtung. Ansonsten würde das Polygon sich entweder selbst schneiden (s. Abb. 3.6 und Abb. 3.8) oder aber die Abtriebsrolle würde die Kurvenscheibe blockieren bzw. sich von ihr lösen (s. Abb. 3.7).

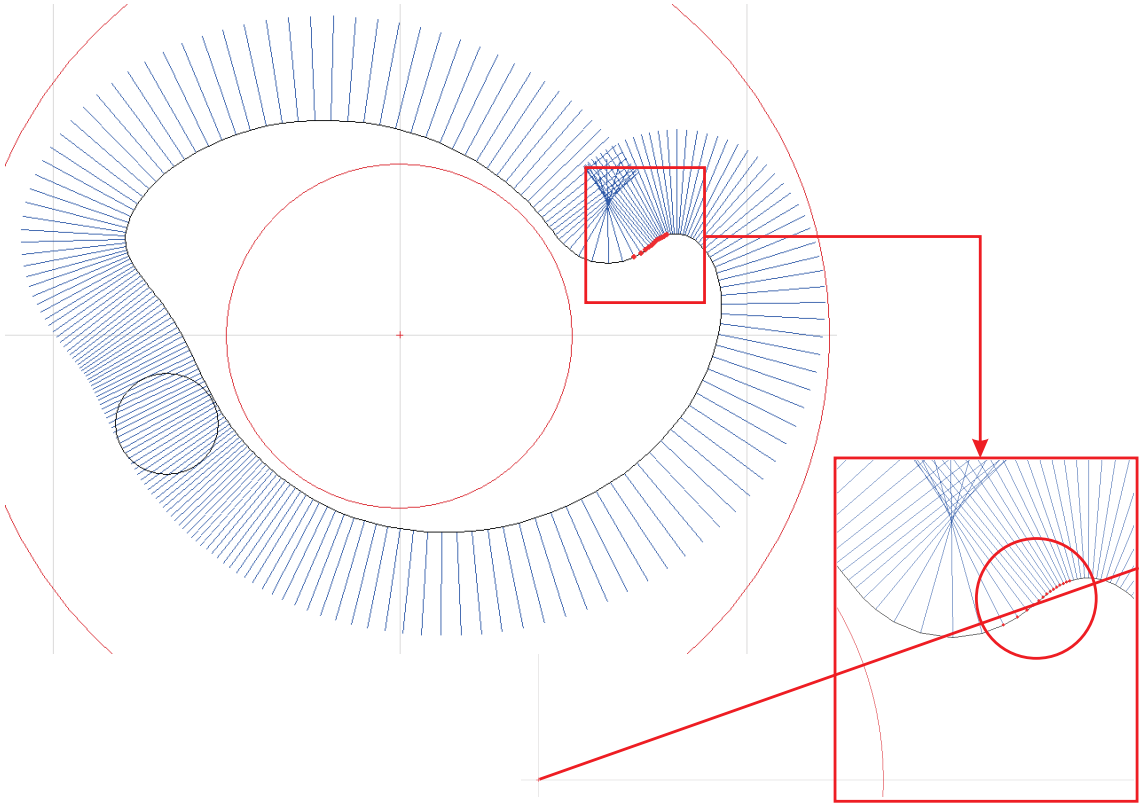


Abbildung 3.7: *Winkel der Stützpunkte nicht streng monoton wachsend bzw. fallend*

3.3.3 Starker Anstieg bei flachem Eingriffsglied

Wie auch bei Abtriebsrollen kann auch bei flachen Eingriffsgliedern ein für die Getriebeparametrisierung zu steiler Anstieg der Übertragungsfunktion zu einem Inkonsistenten Ergebnis führen. In Abb. 3.8 tritt das gleiche Problem auf, wie in Abb. 3.6. Auch hier kann der Fehler durch die Überprüfung der Winkel der Stützpunkte gefunden und visualisiert werden.
Konkave Kurvenscheibenabschnitte, wie in Abb. 3.7 können hier jedoch aufgrund der Berechnungsart der Stützpunkte von vornherein ausgeschlossen werden.

3.3.4 Flaches Eingriffsglied zu kurz

Der berechnete Stützpunkt liegt auf einer Geraden, auf der sich auch das Eingriffsglied befindet. Dessen Position und Länge sind wohlbekannt, jedoch ist nicht von vornher-

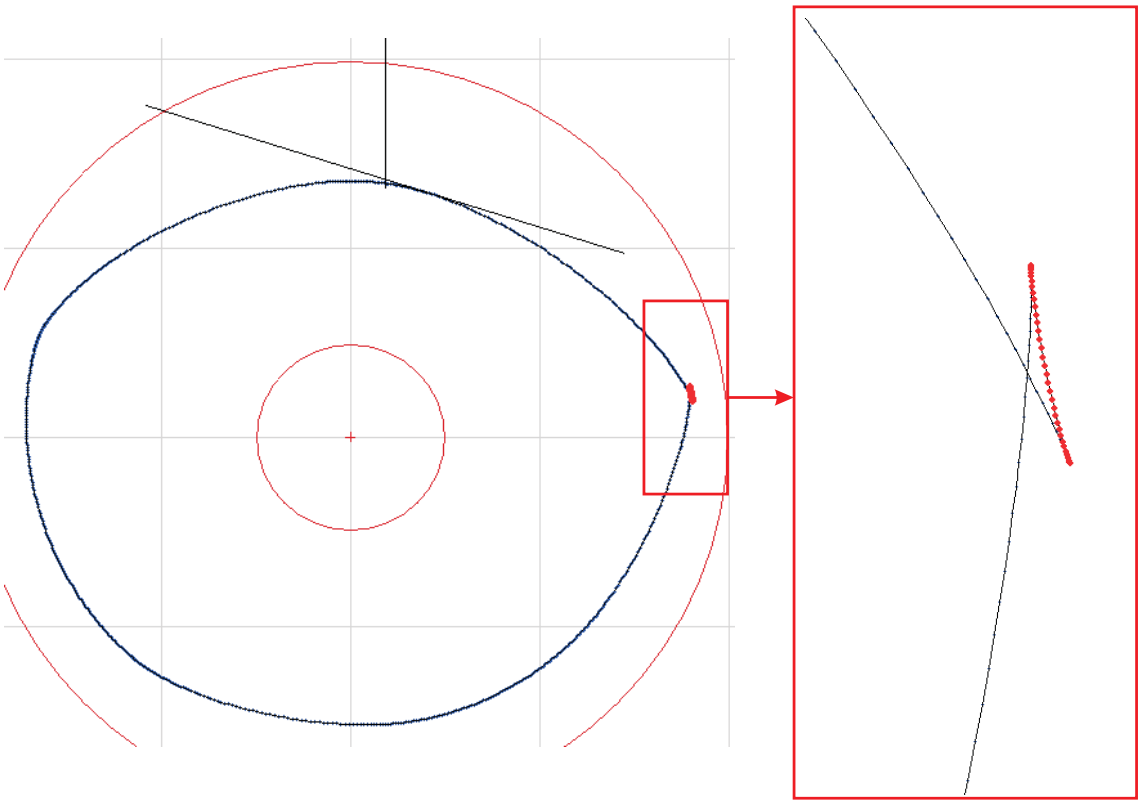


Abbildung 3.8: *Zu steiler Anstieg in der Übertragungsfunktion*

ein sichergestellt, dass der Berührpunkt sich innerhalb der durch das Eingriffsglied beschränkten Strecke befindet. Dieses Problem wird in Abb. 3.9 dargestellt. Dem Problem kann z.B. mit folgenden Varianten begegnet werden:

- Verlängern des Eingriffsgliedes
- Verändern der Position des Abtriebsgliedes
- Verändern des Offsets bei Hebeln
- Verändern des Übertragungswinkels bei Stößeln

Punkt auf einer Strecke

Ob ein flaches Eingriffsglied zu kurz ist kann automatisch erkannt werden, indem geprüft wird, ob sich der Punkt auf der Strecke befindet. Da der Punkt bereits auf

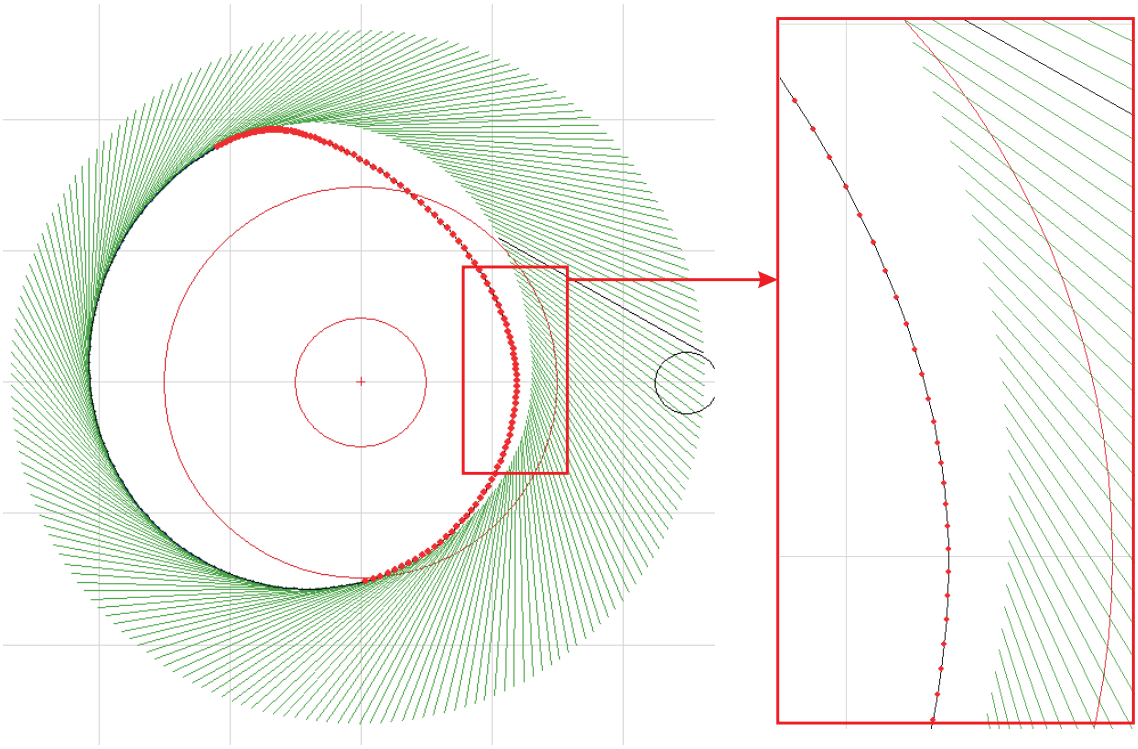


Abbildung 3.9: *Das flache Eingriffsglied ist zu kurz und erreicht die berechnete Kurvenscheibenkontur nicht immer.*

der Geraden liegt, muss lediglich die Entfernung des Punktes vom Mittelpunkt der Strecke berechnet werden. Diese darf maximal die Hälfte der Streckenlänge betragen.

3.3.5 Probleme bei zu geringer Abtastzahl

Je niedriger die Abtastzahl gewählt wird, desto weniger Details des Kurvenverlaufs der Übertragungsfunktion werden berücksichtigt und desto weniger Stützpunkte werden für die Kurvenscheibe berechnet. Um Fehler zu minimieren, kann zwischen den Stützpunkten zwar auf verschiedene Weise interpoliert werden, doch einerseits ist damit nicht sichergestellt, dass diese Interpolation auch im Sinne der definierten Übertragungsfunktion verläuft, andererseits liegen bei vielen Berechnungen oft Polygone zu Grunde, womit auch eine Darstellung der Kurvenscheibe als Polygon ihre Berechtigung findet. Daher soll hier auf ein Problem aufmerksam gemacht werden, dass am Beispiel von Abtriebsrollen besonders deutlich zu sehen ist. In Abb. 3.10 ist eine Übertragungs-

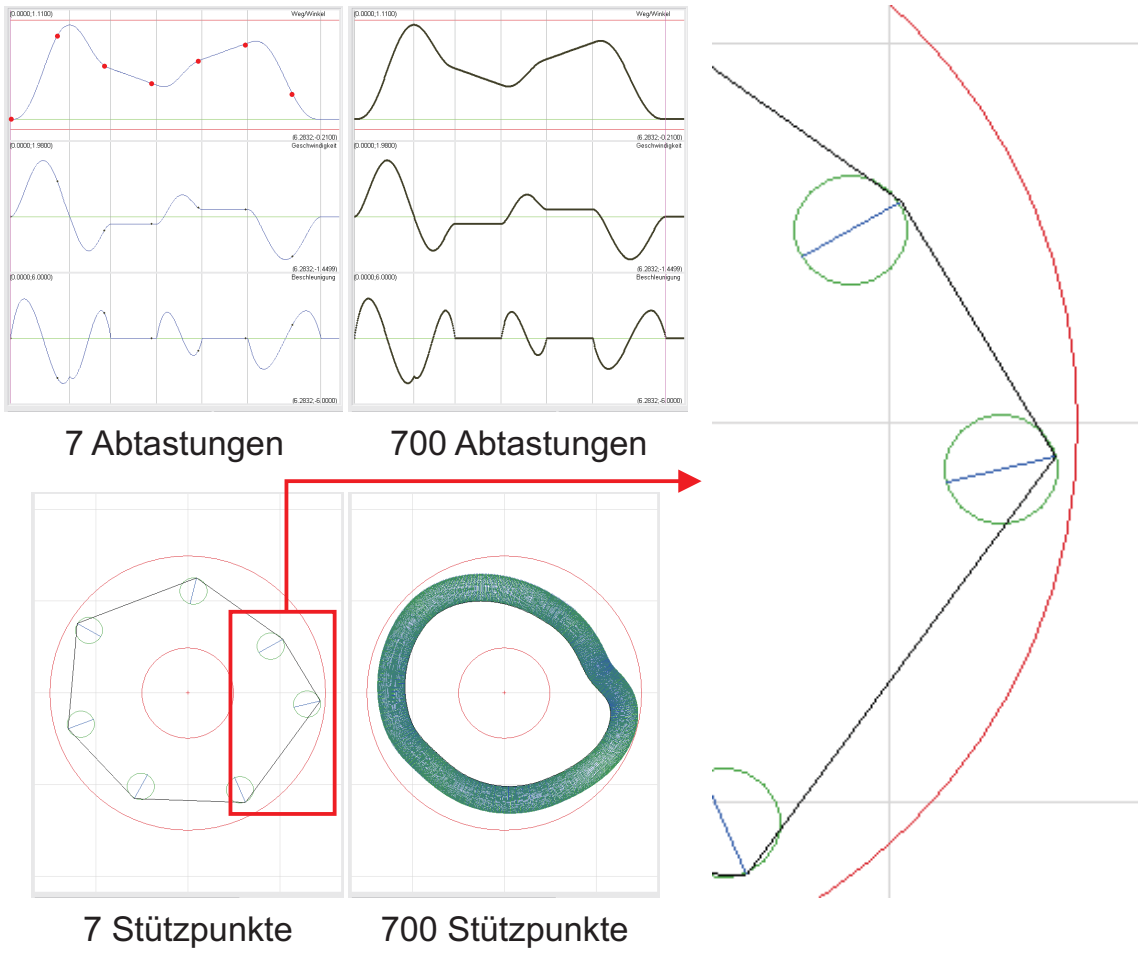


Abbildung 3.10: *Probleme bei zu geringer Abtastzahl*

funktion einmal an 7 Stellen abgetastet worden und ein zweites Mal an 700 Stellen. Die jeweils resultierenden Kurvenscheibenkonturen sind unter den Funktionsverläufen dargestellt. Die berechneten Stützpunkte sind stets korrekt, jedoch kann die Abtriebsrolle im rechts vergrößerten Ausschnitt aufgrund der Kurvenkontur nicht einmal jeden berechneten Punkt erreichen, geschweige denn sich zwischen den berechneten Punkten auf der gewünschten Bahn bewegen.

Derartige Fehler können visualisiert werden, indem eine zweite Kurvenscheibe mit wesentlich höherer Abstastrate zum Vergleich angezeigt wird. Somit kann der Nutzer leichter eine hinreichende Stützpunktzahl für die Kurvenscheibe definieren.

3.4 Animation

Mittels einer Animation lässt sich das Verhalten eines Kurvenscheibengetriebes sehr leicht nachvollziehen. Es bietet sich an, eine solche Vorschau im Getriebeeditor des Kurvenscheibengetriebes zu integrieren. Dort kann sie über einen Schalter aktiviert oder deaktiviert werden. Läuft eine Animation, so wird das Getriebefenster ständig neu gezeichnet. Der Inhalt des Fensters ändert sich mit jeder Neuzeichnung. Eine solche Zeichnung wird als Frame (engl. für Rahmen) bezeichnet. Während der Animation sollte nur das Kurvenscheibengeriebe (Kurvenscheibe und Abtriebsglied) gezeichnet werden. Weitere Zusatzinformationen, wie sie in der statischen Ansicht noch sinnvoll waren (Kurvenscharen, Stützpunkte, ...), können hier die Übersichtlichkeit gefährden. Außerdem muss der Aufwand zur Berechnung eines Frames gering gehalten werden, um eine rasche Bildfolge und somit den Eindruck einer fließenden Bewegung zu ermöglichen. Sämtliche Getriebeparameter können auch während der Animation wie beschrieben verändert werden. Es wird, wenn nötig eine neue Kurvenkontur ermittelt und im nächsten Frame korrigiert dargestellt.

3.4.1 Zwanglaufsicherung der Abtriebsglieder

Ein besonderer Vorteil der Animation besteht darin, Widersprüche oder ungünstige Positionen deutlich zu machen, die sich nicht aus der Form an sich ergeben. Die berechnete Kurvenscheibe ermöglicht dem Abtriebsglied bei ständigem Kontakt, die von der Übertragungsfunktion geforderte Position einzunehmen. Physikalische Zusammenhängen, wie Kräfte, Reibung oder Material werden in die Berechnung nicht einbezogen. Die Motivation des Abtriebsglieds, diese Sollposition zu erreichen, ist damit allein also nicht sichergestellt. Ein besonders deutliches Beispiel ist in Abbildung 3.11 zu sehen. Im linken Bild befindet sich der Stößel stehts oberhalb der x-Achse, also muss eine Kraft nach unten wirken, um ihn zu motivieren, an der Kurvenscheiben zu bleiben. Im rechten Bild, gelangt er jedoch auch in den Bereich unterhalb der x-Achse, wo eine Kraft nach oben wirken müsste.

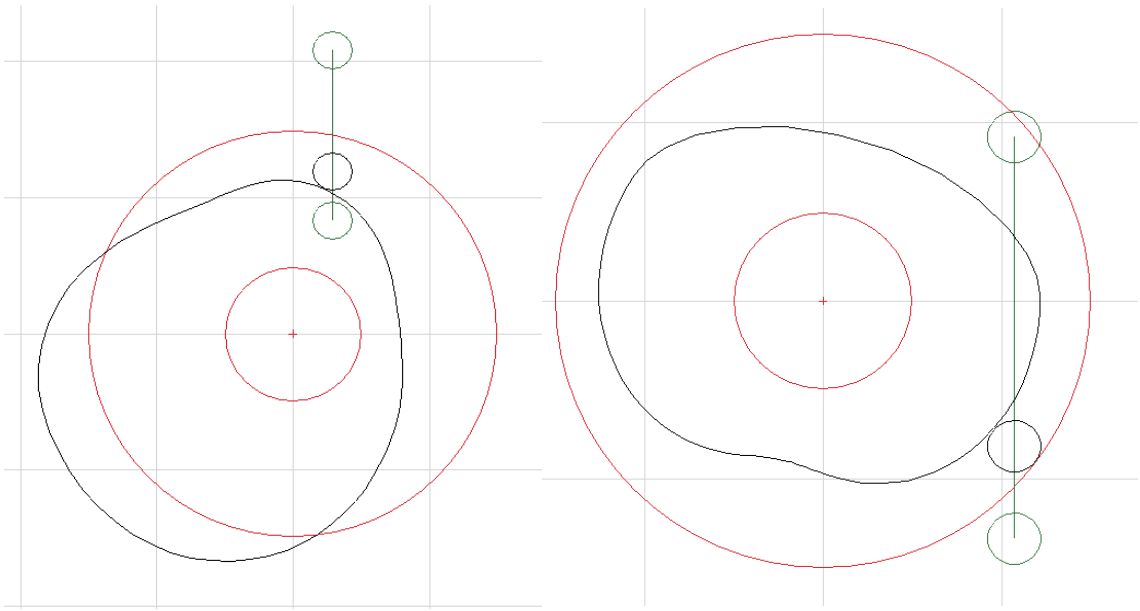


Abbildung 3.11: *Zwanglaufsicherung von Abtriebsgliedern: Rollenstößel. Da die Kurvenscheibenkontur auf Basis geometrischer Zusammenhänge berechnet wird, wird die Zwanglaufsicherung dadurch allein nicht sichergestellt.*

Kapitel 4

Softwaretechnische Realisierung

4.1 Werkzeuge

4.1.1 Entwicklungsumgebung

Die Entscheidung, welche Entwicklungsumgebung (IDE) der Nutzer in welchem Betriebssystem verwendet, ist abhängig von den Anforderungen der verwendeten Bibliotheken. Im Idealfall existieren für jede Bibliothek entsprechende Projektdateien für die gängigen IDE's in den verschiedenen Betriebssystemen. Leider ist dies oft nicht der Fall und der Entwickler hat die Wahl, die Konfiguration der Bibliothek an die eigene Umgebung anzupassen oder die Umgebung zu wechseln. Für dieses Projekt empfiehlt sich die Verwendung von Microsoft Visual Studio 2003 [10]. Hauptgrund hierfür ist, dass auch das für den Export verwendete MASP-SDK [11] in dieser Umgebung entwickelt wurde und hier am einfachsten kompiliert werden kann.

4.1.2 Programmiersprache

Grundsätzlich ist die Realisierung eines solchen Projekts in den verschiedensten Programmiersprachen möglich. Die Entscheidung fiel hier auf C++, da es sich so leichter an andere Projekte, die ebenfalls in C++ umgesetzt wurden, anpassen lässt. Ein weiterer Grund ist der gewünschte Export als MASP-Datei, wofür ein SDK [11] auf C++-Basis geschaffen wurde.

4.1.3 Grafische Oberfläche

Es existiert eine Reihe von Bibliotheken, die zur Gestaltung einer grafischen Nutzeroberfläche genutzt werden können. In die nähere Auswahl für dieses Projekt kamen Qt [15] und Fox [5] aufgrund von Entwicklungsstand, Handhabbarkeit, Portierbarkeit und der freien Verfügbarkeit. Letztenendes fiel die Entscheidung auf das Fox Toolkit, da es lizenztechnisch weniger problematisch ist. Es steht eine Projektdatei für Visual Studio 6 zur Verfügung, welche sich für Visual Studio 2003 konvertieren lässt.

4.1.4 XML

Für den Umgang mit XML-basierten Konfigurationsdateien wurde der Xerces-C++ Parser [1] verwendet, der auch für das MASP-SDK benötigt wird.

4.1.5 Mathematische Bibliotheken

C++-Bibliotheken, die symbolisches Rechnen erlauben, gibt es zwar mehrere, doch die meisten sind nicht kostenfrei verfügbar. GiNaC [6] ist eine Bibliothek, die nach der GPL genutzt werden kann. Die Bibliothek ist von der ebenfalls frei verfügbaren Bibliothek CLN [7] abhängig. GiNaC wurde mit GCC entwickelt und die Dokumentation bezieht sich ausschließlich darauf. Für die Verwendung unter Windows empfiehlt sich g++ in Cygwin zu nutzen. Leider ist mit GiNaC bislang z.B. keine symbolische Integration möglich.
Letztendlich wurde entschieden, lediglich die Standard-Bibliothek *math* zu nutzen und kompliziertere Berechnungen, wie das Differenzieren von Funktionen oder Lösen linearer Gleichungssysteme während der Implementierung allgemein zu lösen und die Ergebnisse dem Programm direkt zur Verfügung zu stellen. Dadurch reduziert sich der Berechnungsaufwand zur Laufzeit, was auch Geschwindigkeitsvorteile mit sich bringt.

rechnerische Genauigkeit

Es stellt sich die Frage, ob die vorgestellten Verfahren unter Verwendung von Standarddatentypen hinreichend genau arbeiten. Der Prototyp wurde unter Microsoft Visual

C++ [10] erstellt. Der zugehörige Compiler sieht für den Datentyp ***long double*** ebenso wie für den Datentyp ***double*** lediglich acht Bytes vor. Insbesondere an zwei Stellen kann es zu numerischen Problemen kommen. Dabei handelt es sich zum einen um die Koeffizientenermittlung des Polynoms 5. Grades (s. Abschnitt A) und zum anderen um die numerische Berührpunktermittlung (s. Abschnitt 2.3.5 u. 2.3.6).
Das Problem bei der Koeffizientermittlung besteht darin, dass bei einigen Rechenschritten gültige Stellen verloren gehen, wenn die Distanz zweier Abschnittsgrenzen sehr gering ist. Dieses Problem kann umgangen werden, indem der Nenner in den von Derive [13] gelieferten Gleichungen, wie in Abschnitt A ist beschrieben, ersetzt wird.
Bei der numerischen Berühpunktermittlung stellt sich das Problem am Beispiel eines flachen Eingriffsglieds wie folgt dar. Es muss der theoretisch infinitesimale Nachbar einer Gerade in der Kurvenschar ermittelt werden. Je näher dieser Nachbar ist, desto höher ist die Wahrscheinlichkeit, dass diese Geraden nahezu parallel liegen. Liegen zwei Geraden parallel, liegt der Schnittpunkt im Unendlichen und der Berührpunkt kann nicht bestimmt werden. Um Artefakte durch Rechenfehler zu vermeiden, kann es sinnvoll sein den Abstand zur angenommenen infintesimalen Nachbargeraden von $\epsilon = 10^{-8}$ auf beispielsweise 10^{-5} (im Bogenmaß der Kurvenscheibe) zu reduzieren. Dieses Vorgehen ist für Kurvenscheiben mit einer Stützpunktzahl bis 100000 empfehlenswert. Eine Alternative hierzu stellt die Nutzung genauerer Datentypen dar. Zum einen bieten andere Compiler für ***long double*** eine höhere Genauigkeit, zum anderen existieren mathematischen Bibliothteken [14] [2] [7], die weitaus genauere Rechnungen ermöglichen.

4.2 Mathematische Datentypen/Klassen

Für ein leichteres Verständnis mathematischer Berechnungen und Zusammenhänge werden neben den üblichen C++-Datentypen einige Klassen definiert. Abbildung 4.1 zeigt Name und Attribute der wesentlichsten Klassen.

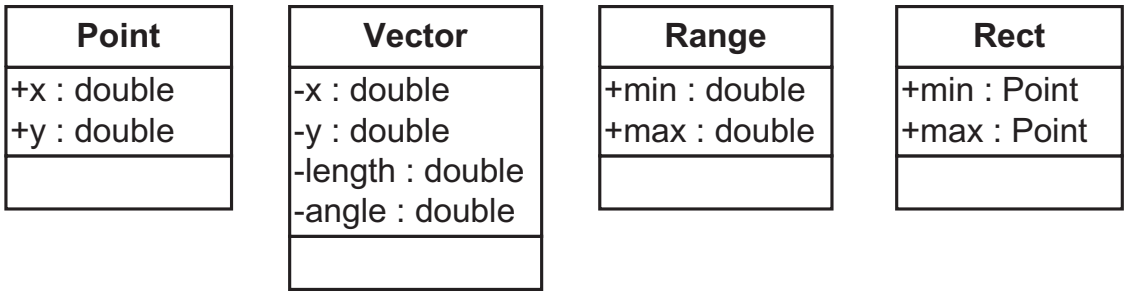


Abbildung 4.1: *mathematische Datentypen/Klassen*

Point

Ein *Point* (Punkt) wird in kartesischen Koordinaten durch einen x- und einen y-Wert definiert.

Vector

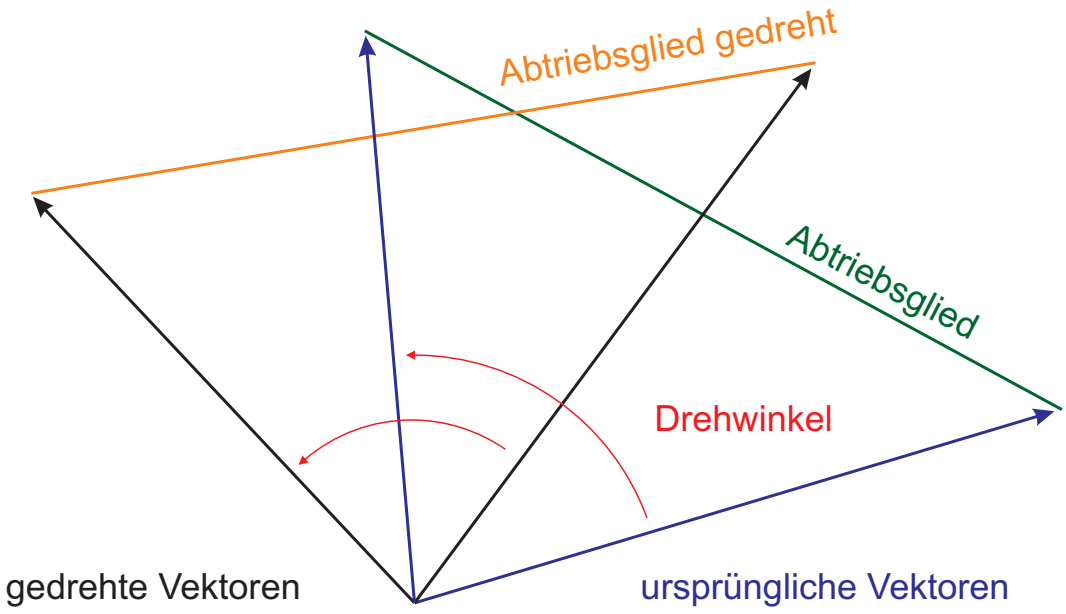


Abbildung 4.2: *Nutzung von Vektoren für Rotationen*

Ein *Vector* stellt einen in der Ebene liegenden Vektor dar. Er kann in Polarkoordinaten über seinen Betrag *length* und seinen Winkel *angle* definiert sein oder aber in kartesischen Koordinaten über x und y. Diese Attribute werden gleichzeitig gehalten

und das Verändern eines dieser Attribute führt zu einer Neuberechnung zweier anderer. In diesem Projekt finden Vektoren vor allem Anwendung bei Rotationen. Wenn beispielsweise die Kurvenschar des Abtriebsglieds um den zugehörigen Antriebswinkel der Kurvenscheibe rotiert dargestellt werden sollen, müssen die ursprünglichen Koordinaten, die das Abtriebsglied beschreiben, lediglich in Vektoren konvertiert werden, zu deren Winkel dann der Winkel der Kurvenscheibe addiert wird (s. Abb. 4.2). Der Vorteil, der sich durch das Legen des Koordinatenursprungs auf das Rotationszentrum der Kurvenscheibe ergibt, wird hier besonders deutlich.

Range

Eine *Range* kennzeichnet einen eindimensionalen Bereich. Damit können beispielsweise *min*- und *max*-Werte einer Variablen zusammengefasst werden. Außerdem lässt sich einfach prüfen, ob ein zulässiger Bereich überschritten wurde.

Rect

Ein *Rect* (Rechteck) definiert sich über zwei gegenüberliegenden Eckpunkte. Es ist vor allem bei Koordinatentransformationen (s. Abb. 4.3) nützlich, bei denen zwischen dem Koordinatensystem von Übertragungsfunktion bzw. Getriebe und den Bildschirmkoordinaten umgerechnet werden muss.

coordTransform

Aufgrund ihrer häufigen Nutzung im Projekt soll an dieser Stelle auch die Methode *coordTransform* Erwähnung finden. Mit dieser Funktion können die kartesischen Koordinaten eines Punktes in einem anderen Koordinatensystem ermittelt werden. Hierzu müssen zwei ein Rechteck aufspannende Punkte in beiden Koordinatensystemen bekannt sein. Dann kann zu jedem weiteren Punkt im ursprünglichen Koordinatensystem die Äquivalenz im Zielkoordinatensystem ermittelt werden (s. Abb. 4.3). Die Hauptanwendung liegt in der Umrechnung von Koordinaten der Übertragungsfunktion in Bildschirmkoordinaten oder auch Getriebekoordinaten in Bildschirmkoordinaten. Außerdem ist es mit dieser Methode möglich, den Kurvenverlauf eines Abschnitts zu

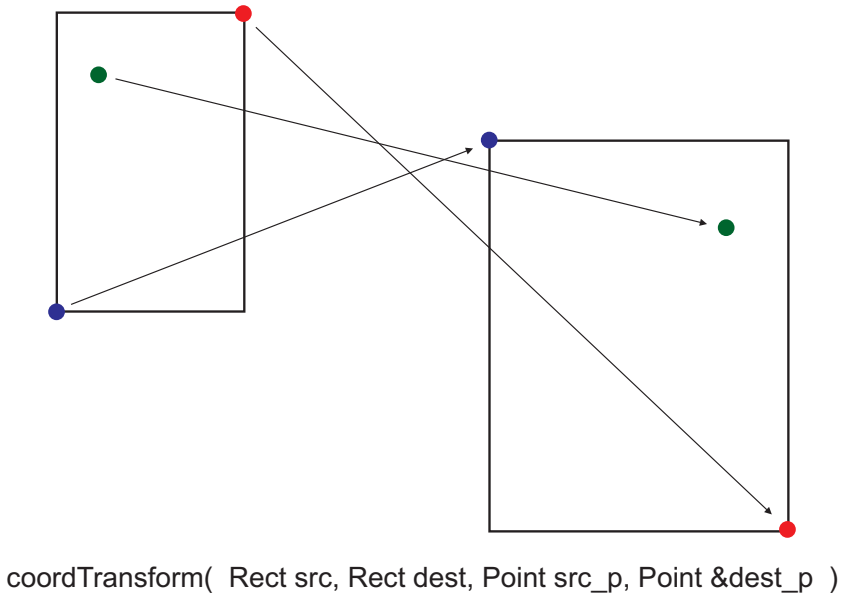


Abbildung 4.3: *Anwendung von coordTransform*

normalisiern bzw. einen normalisierten Kurvenverlauf auf die tatsächlichen Koordinaten zu strecken. Das Rechteck für die normalisierten Koordinaten ist dann durch die Eckpunkte $(0;0)$ und $(1;1)$ definiert, während die äquivalenten realen Koordinaten $(x_0;y_0)$ und $(x_1;y_1)$ betragen. Hierbei ist es durchaus zulässig, dass y_1 kleiner als y_0 ist, wodurch die Funktion real abfällt, während die normierte Funktion steigt. Zusätzlich kann dieselbe Methode auch verwendet werden, um eine eindimensionale Transformation zu berechnen, indem von den anzugebenden Koordinaten lediglich die x oder y-Werte berücksichtigt werden.

4.2.1 Einzeichnen der Übertragungsfunktion

Am Beispiel des Einzeichnens der Übertragungsfunktion (s. Abb. 4.4) soll die Anwendung von $coordTransform$ weiter vertieft werden. Die Klasse für die Übertragungsfunktion stellt Methoden bereit, die für einen beliebigen x-Wert den zugehörigen Funktionsabschnitt ermitteln und in diesem den Funktionswert oder eine Ableitung davon an der Stelle x zurückgeben können. Soll die Funktion vollständig auf den Bildschirm dargestellt werden, so dienen die Bildschirmkoordinaten als Grundlage. Die Ansichtsparameter $minXView$ und $maxXView$ beschreiben den im Fenster darzu-

stellenden Bereich für x-Werte. Sei die Fenstergröße mit *width* und *height* angegeben, so veranschaulicht folgender Programmcode die Ermittlung der einzuzeichnenden Punkte:

```
Rect windowBorder(Point(0,height),Point(width,0));
Rect viewBorder(Point(minXView,minYView),Point(maxXView,maxYView));
for ( int i = 0; i < width; i++ )
{
    Point wPoint( i, 0 );
    Point rPoint;
    coordTransform( windowBorder, viewBorder, wPoint, rPoint );
    rPoint.y = getY( rPoint.x );
    coordTransform( viewBorder, windowBorder, rPoint, wPoint );
    draw( wPoint );
}
```

Zu beachten ist zunächst, dass die vertikale Achse der Bildschirmkoordinaten nach unten zeigt und am oberen Bildschirmrand ihren Ursprung hat. Als Referenzpunkt dient hier jedoch die linke untere sowie die rechte obere Ecke des Fensters. Der Punkt $wPoint$ beschreibt Fensterkoordinaten und der Punkt $rPoint$ Koordinaten der Übertragungsfunktion. Da die Funktion über das komplette Fenster eingezeichnet werden soll, ist zunächst nur der gewünschte x-Wert in Fensterkoordinaten bekannt. Über $coordTransform$ wird der äquivalente x-Wert in Funktionskoordinaten berechnet. Zu diesem wird mit $getY$ der zugehörige y-Wert ermittelt. Im Anschluss wird der Punkt wieder in Fensterkoordinaten umgerechnet, um den y-Wert in Fensterkoordinaten zu erhalten. Für jeden x-Wert in Fensterkoordinaten kann so der passende y-Wert ermittelt werden. Darstellungsfehler durch steile Anstiege in Fensterkoordinaten können vermieden werden, indem zwei benachbarte ermittelte Punkte durch Linien verbunden werden.
Sollen lediglich die Abstastwerte eingezeichnet werden (s. Abb. 4.4), verkürzt sich der Rechenweg, da hier die Koordinaten der Übertragungsfunktion die Grundlage bilden. Der Definitionsbereich sei durch $x.min$ und $x.max$ beschrieben, während dx den Ab-

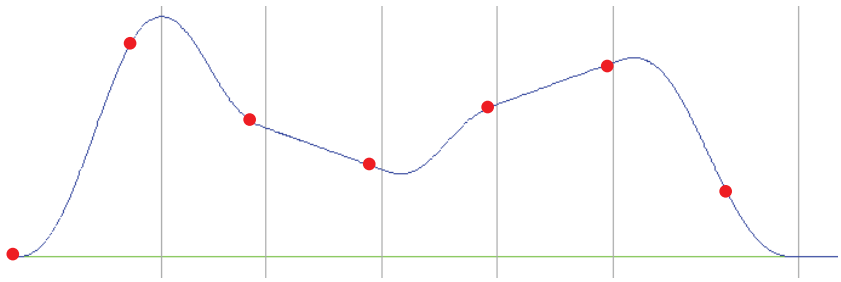


Abbildung 4.4: *Einzeichnen von Kurvenverlauf (durchgehende Linie) und Abtastwerten (hier 7 Punkte)*

stand zwischen zwei Abtastwerten beschreibt:

```
Rect windowBorder(Point(0,height),Point(width,0));
Rect viewBorder(Point(minXView,minYView),Point(maxXView,maxYView));
for ( double i = x.min ; i < x.max ; i += dx )
{
    Point rPoint( i, getY( i ) );
    Point wPoint;
    coordTransform( viewBorder, windowBorder, rPoint, wPoint );
    draw( wPoint );
}
```

4.3 Parameter

4.3.1 Einheiten

Zur Vereinheitlichung der Repräsentation der Daten werden alle Winkel im Bogenmaß angegeben. Die Maßeinheit, die genutzt wird um eine Position oder Länge zu beschreiben, sei einheitlich mit LE bezeichnet. Der Nutzer kann diese Längeneinheit ohne Beschränkung der Allgemeinheit mit einer beliebigen Einheit substituieren. Dabei ist zu Beachten, dass einige Parameter und selbst die Werte der Übertragungsfunktion abhängig vom gewählten Getriebetyp als Winkel oder als Position zu interpretieren sind.

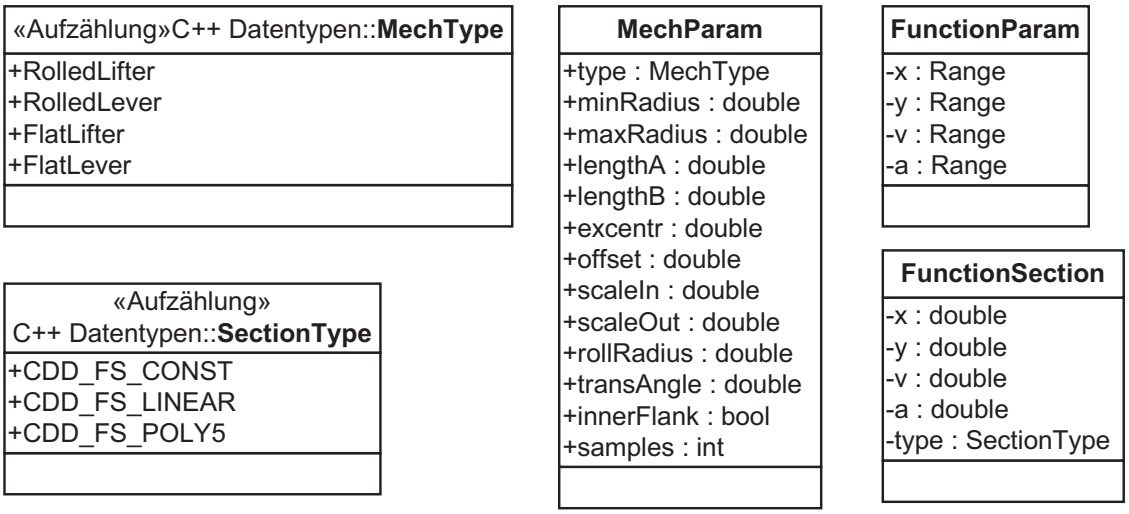


Abbildung 4.5: *Parameter für Übertragungsfunktion, Getriebe und Funktionsabschnitt*

4.3.2 Übertragungsfunktion

Aus softwaretechnischer Sicht kann die Übertragungsfunktion als doppelt verkettete Liste interpretiert werden. Ein Element dieser Liste enthält die Parameter für eine Grenze zweier Funktionsabschnitte sowie den Kurvenverlaufstyp zur nächsten Grenze (s. Abb. 4.5). Die Liste ist nach den x-Werten der Abschnittsgrenzen sortiert. Eine Implementierung der Liste ist beispielsweise möglich, indem jedes Element Zeiger auf die Nachbarelemente erhält. Alternativ kann man auch mit *std :: list <FunctionSection>* und zugehörigen Iteratoren arbeiten. Die Bearbeitung vor allem von Sonderfällen unterscheidet sich hierbei. Das nächste Element des Elements mit dem größten x-Wert, ist das Element mit dem kleinsten x-Wert und umgekehrt. Für die Verlaufsberechnung des Abschnitts zwischen diesen Elementen werden zwei Parameter der Funktionsdefinition benötigt: $x.min$ und $x.max$.

Erweiterung

Sollen neue Typen implementiert werden, die den Kurvenverlauf eines Abschnitts beschreiben, sind Namen für diese in *SectionType* zu ergänzen. Sollten diese Verlaufstypen mehr Parameter benötigen, als x, y, v, a an beiden Abschnittsgrenzen, so sind diese in

FunctionSection zu integrieren. Es ist durchaus zulässig, dass ein Verlaufstyp nicht alle in den Abschnittsgrenzen angegebenen Parameter berücksichtigt. Natürlich müssen auch die Berechnungsvorschriften für den Verlaufstyp integriert werden. Dies spielt für den Export in Konfigurationsdateien jedoch keine Rolle.

4.3.3 Getriebe

Um das Getriebe über Parameter zu beschreiben, ist zunächst der Getriebetyp festzulegen (s. Abb. 4.5). Es existieren eine Reihe von Parametern, die die gleiche oder ähnliche Bedeutung in anderen Getriebetypen haben, andere Parameter werden komplett anders interpretiert und wieder andere finden in bestimmten Getriebetypen keine Bedeutung. *minRadius* und *maxRadius* beschreiben bespielsweise die zugelassene Grenzwerte für die Dimension der Kurvenscheibe. Für die Berechnung der Kurvenkontur selbst spielen diese Werte jedoch eine untergeordnete Rolle. Der *offset* beschreibt die Ausgangsstellung des Eingriffsglieds und wird je nach Getriebetyp anders interpretiert. So stellt der *offset* bei Hebeln einen Winkel dar, während es bei Stößeln ein Weg beschreibt. Der *rollRadius* wird bei Getriebetypen ohne Abtriebsrolle ignoriert. Durch das Halten sämtlicher Getriebeparameter ist ein schneller Wechsel zwischen den Getriebetypen möglich.

Erweiterung

Um andere Getriebetypen zu definieren, sind Namen dafür in *MechType* zu ergänzen. Bereits vorhandene Parameter können genutzt werden und sollten es, wenn deren Bedeutung nicht widersprüchlich zu deren Namen erscheint. Bei Bedarf können neue Parameter ergänzt werden, die dann von bisherigen Getriebetypen ignoriert werden. Mit der bisherigen Definition ist es allerdings nur möglich Kurvenscheibengetriebe mit einem Eingriffsglied und einer Kurvenkontur zur erzeugen. Wie diese Grenzen zu überwinden sind, ist in Abschnitt 4.7 beschrieben.

4.4 Kurvenscheibensynthese

Bei der Synthese der Kurvenscheibe werden zunächst die Antriebsdrehwinkel berechnet, für die ein Stützpunkt der Kurvenscheibe berechnet werden soll. Der Abstand zwischen diesen Winkeln beträgt $(x.max - x.min)/samples$. Für den jeweiligen x-Wert muss nun der zugehörige Funktionsabschnitt ermittelt werden und daraufhin der Funktionswert und ggf. Ableitungen davon an der Stelle x. Jetzt wird die Lage des Eingriffsglieds berechnet sowie der Berührpunkt mit der Kurvenscheibe. Dieser Berührpunkt mit der Kurvenscheibe muss anschließend noch gegen den Antriebsdrehwinkel gedreht werden. Das Ergebnis ist ein Stützpunkt der Kurvenscheibe.

4.5 Animation

Die gewählte Anzahl der Frames für eine Umdrehung der Kurvenscheibe sei n und die laufende Nummer des aktuell zu zeichnenden Frames sei i. Überschreitet i n, so kann es auf $i \mod n$ zurückgesetzt werden. Liegt die korrekte Kurvenscheibenkontur vor, so muss sie mit jedem Frame um $dx = 2 \cdot \pi/n$ weitergereht werden. Für das Abtriebsglied wird der y-Wert der Übertragungsfunktion für den aktuellen Drehwinkel bestimmt. Damit kann die Lage des Abtriebsglieds ermittelt werden und das Getriebe kann gezeichnet werden.
Im Prototyp des Kurvenscheibeneditors wurden Fenster verwendet, deren Methode zum Zeichnen (*paint()*) aufgerufen wird, sobald sie eine Nachricht mit der Forderung zur Aktualisierung (*update()*) erhalten. Um automatisch das Zeichnen des nächsten Frames auszulösen, muss dieses *update()* am Ende von *paint()* aufgerufen werden. Ohne zusätzliche Maßnahmen, wäre der Rechner nun mit dem Zeichnen der Animation voll ausgelastet. Damit die Anwendung zwischen den Frames genügend Zeit hat, auf Eingaben zu reagieren muss *paint()* pausieren. Um das zu erreichen, kann am Ende von *paint()* ein separater Thread gestartet werden. Der Anwendung stehen nun wieder genügend Ressourcen zur Verfügung, um alle Nachrichten rechtzeitig abzuarbeiten und auf Eingaben zu reagieren. Gleichzeitig macht der separate Thread nichts weiter, als eine gewisse Zeit zu warten, um danach *update()* für das Fenster aufzurufen, woraufhin

das Fenster bei nächster Gelegenheit neu gezeichnet wird.
Nachteilig bei dieser Methode ist, dass die Zeit zwischen zwei Frames nicht exakt definierbar ist. Sie ist zum einen abhängig von der Zeit, die zum Berechnen der darzustellenden Grafiken benötigt wird, und zum anderen von der Abarbeitung der Nachrichten im System. Zumindest die Zeit für die Berechnungen kann kompensiert werden. Dazu wird der separate Thread erweitert. Er stellt zu Beginn die Uhrzeit fest und berechnet anschließend den nächsten Frame. Jetzt wird nur noch die Differenz zur Sollzeit abgewartet, bevor *update()* aufgerufen wird.

4.6 Schnittstellen

4.6.1 Konfigurationsdateien

XML bietet eine ideale Grundlage zur Speicherung der Konfiguration eines Getriebes. Das beinhaltet sowohl sämtliche Parameter als auch die Abschnitte der Übertragungsfunktion. Zusätzlich können programmspezifische Informationen wie die Version oder die Parameter zur Betrachtung von Übertragungsfunktion und Getriebe definiert werden. Mit solchen Konfigurationsdateien kann die Entstehung eines Kurvenscheibengetriebes auf anderen Systemen nachvollzogen werden ohne im Programm zusätzliche Einstellungen vorzunehmen.

4.6.2 MASP

Das MASP-SDK [11] bietet Routinen, die die Konstruktion gültiger Getriebe im MASP-Format erleichtern. Somit ist es möglich Kurvenscheiben zu erstellen, indem das Rotationszentrum sowie die Stützpunkte übergeben werden. Das Abtriebsglied ist aus anderen elementaren Bausteinen zusammenzusetzen. Über die Definition von Randbedingungen (engl. Constraints) lässt sich das Verhalten des Abtriebsglieds nachbilden. Beim Export von Stößeln ins MASP-Format ist insbesondere zu beachten, dass sich konstruktionsbedingt der Stößel sich nur in dem Bereich bewegen kann, der mit den Funktionsparametern $y.min$ und $y.max$ definiert wurde. Der Grund hierfür ist,

dass die Bahn, auf der sich der Stößel bewegen kann, durch eine Strecke fester Länge definiert wird. Für Hebel gilt diese Einschränkung nicht.

4.6.3 Stapelverarbeitung

Die Nutzung der Kommandozeile unter Ausschluss einer grafischen Oberfläche für die Erstellung von Kurvenscheiben ist nur dann sinnvoll, wenn vorher sichergestellt wurde, dass das Getriebe richtig ausgelegt ist. Liegt eine entsprechende Sammlung von Konfigurationsdateien vor, so können aus diesen auf diese Weise MASP-Dateien oder Dateien eines anderen Formats generiert werden.

4.7 Formschlüssige Kurvenscheibengetriebe

Die durch den vorgestellten Kurvenscheibeneditor erzeugten Kurvenscheiben unterstützen nur eine Kurvenkontur und nur ein Eingriffsglied. Das suggeriert die Notwendigkeit einer kraftschlüssigen Zwanglaufsicherung. Verschiedene Ausgaben des Editors (Kurvenscheiben inkl. Abtriebsglied) lassen sich jedoch unter bestimmten Umständen kombinieren, sodass die Zwanglaufsicherung durch Formpaarung erfolgt. Beispiele hierfür werden im Folgenden erläutert.

4.7.1 Nutkurvenscheibe mit Abtriebsrollen

Um eine Nutkurvenscheibe zu erzeugen, in der sich eine Abtriebsrolle bewegt, werden zwei Getriebe erzeugt, die sich ausschließlich in der genutzten Flanke der Abtriebsrolle unterscheiden. Die zweite Kurvenscheibe kann dann in das zuerst erzeugte Getriebe integriert werden. Um Problemen vorzubeugen empfiehlt sich eine sehr hohe Abtastzahl und eine nachträgliche Verringerung des Rollenradius. Auf diese Weise können Engpässe für die Rolle vermieden werden.

4.7.2 Kurvenscheiben mit zwei Abtriebsrollen

Sowohl Nut- als auch Wulstkurvenscheiben für zwei Abtriebsrollen können durch Zusammenschaltung der Ausgabe des Editors erzeugt werden. Die Abtriebsglieder sind dabei starr miteinander verbunden.

Zwei Rollenstößel

Um eine zweite Kurvenscheibe für dieselbe Übertragungsfunktion zu erzeugen, kann beim zweiten Rollenstößel der Offset, die Exzentrizität sowie der Rollenradius beliebig verändert werden. Natürlich sollte für eine Zwanglaufsicherung auch die entgegengesetzte Flanke der Kurvenscheibe genutzt werden. Die Gemeinsamkeit besteht in der Bewegungsrichtung sowie der Abtriebswertskalierung, womit beide Rollen starr verbunden werden können. Es muss dabei darauf geachtet werden, dass die zwei Rollenbahnen sich nicht berühren oder gar kreuzen.

Zwei Rollenhebel

Für die Verwendung von zwei Rollenhebeln, muss beim zweiten Rollenhebel der Offset verändert werden. Weiterhin dürfen sich die Länge des Hebels, Exzentrizität sowie der Rollenradius unterscheiden. Beide Hebel teilen sich ein gemeinsames Rotationszentrum sowie Abtriebswertskalierung, wodurch auch diese starr miteinander verbunden werden können.

4.7.3 Integration im Editor

Für eine komfortable Zusammenschaltung zweier Abtriebsglieder und ihrer Kurvenscheiben sind folgende Änderungen im Getriebeeditor nötig:

- Ein zweiter Satz Getriebeparameter muss auswählbar sein.
- Eine zweite Kurvenscheibe muss gespeichert werden.
- Je nach Getriebetyp müssen entsprechende Parameter der beiden Sätze aneinander gebunden werden, sodass diese in beiden Parametersätzen identisch bleiben.

- Sämtliche Berechnungen müssen für beide Eingriffsglieder durchgeführt werden.
- Alle visuellen Elemente sollten für beide Parametersätze separat zu- und abschaltbar sein.
- Eine zusätzliche Validierung sollte integriert werden, die Überschneidungen erkennt. Solche Überschneidungen können dennoch sinnvoll sein, wenn das Getriebe dreidimensional realisiert wird.
- Die Exportfunktionen müssen erweitert werden.

4.8 Effizienz

Um sicherzustellen, dass der vorgestellte Editor die nötige Leistung auf aktuellen Systemen bringt, wurden mit dem Prototypen ein paar Szenarien auf verschiedenen Rechnern durchgeführt. Bei den Szenarien 1 bis 3 wurde die Systemzeit vor und nach den Berechnungen und dem Einzeichnen programmintern gemessen und die Differenz bestimmt. Bei Szenario 4 wurde getestet, ob der Rechner noch über ausreichend Ressourcen verfügt, um auf Eingaben schnell genug zu reagieren und um alle Fensterelemente bei Bedarf neu zu zeichnen.

4.8.1 Testsysteme

Die vier Szenarien wurden auf vier Rechnern durchgeführt. Die Konfiguration der Testsysteme kann in Tabelle 4.1 eingesehen werden.

4.8.2 Laufzeiten

Szenario 1

Im Getriebeeditor wird eine Kurvenscheibe mit einer Stützpunktzahl von 100.000 berechnet und in das Fenster eingezeichnet.

	System 1	System 2	System 3	System 4
CPU	Intel Core2Duo	AMD AthlonXP	Intel Pentium M	AMD Athlon
	2 x 2,13 GHz	1,83 GHz	1,6 GHz	952 MHz
RAM	DDR2-800, 2GB	DDR-333, 2GB	DDR2-533, 1GB	DDR-333, 1GB
GPU	Radeon X1650XT	Radeon 9500Pro	M. Radeon X700	GeForce2 MX
HDD	7200 rpm	7200 rpm	4200 rpm	5400 rpm
OS	WinXP SP2	WinXP SP2	WinXP SP2	WinXP SP2

Tabelle 4.1: *Testsysteme*

	System 1	System 2	System 3	System 4
Szenario 1	0,969s	1,088s	1,093s	2,113s
Szenario 2	6,653s	8,234s	7,656s	19,411s
Szenario 3	0,656s	0,813s	0,781s	2,374s
Szenario 4	ok	ok	ok	ok

Tabelle 4.2: *Auswertung der Laufzeiten*

Szenario 2

Wie Szenario 1. Zusätzlich werden auch Stützpunkte, Kurvenscharen, Abtriebsglied und Fehler ermittelt und eingezeichnet.

Szenario 3

Wie Szenario 2. Die Stützpunktzahl wird auf 10.000 reduziert.

Szenario 4

Animation des Kurvenscheibengetriebes mit 3.600 Stützpunkten.

Auswertung

Aus den Tests ging hervor (s. Tab. 4.2), dass der Prototyp auf allen aktuellen Systemen eine ausreichende Leistung aufweist. Da keine Optimierung der Leistung vorgenommen wurde, sind weitere Leistungssteigerungen möglich.

Speicherbedarf

Der Bedarf des Prototypen an Arbeitspeicher überschritt während der Tests zu keinem Zeitpunkt 16 MB, und wird gegenüber dem Bedarf an Rechenleistung als unwesentlich angesehen. Der Fesplattenspeicher der Anwendung selbst beträgt inklusive der zusätzlich benötigten DLL-Dateien weniger als 3 MB. Eine in das MASP-Format exportiertes Kurvenscheibengetriebe mit 100000 Stützpunkten ist etwa 13 MB groß.

Kapitel 5

Zusammenfassung und Ausblick

Für die gezielte Auslegung von Kurvenscheibengetrieben wurde ein Werkzeug konzipiert. Anhand einer Übertragungsfunktion sowie der Getriebeauslegung werden Stützpunkte einer Kurvenscheibenkontur ermittelt, die eine Umsetzung der Übertragungsfunktion zulassen. Dazu wurden Editoren für die Übertragungsfunktion sowie für die Getriebeauslegung beschrieben, die sich sehr einfach und intuitiv bedienen lassen und Auswirkungen von Veränderungen sowie Fehler in Echtzeit visualisieren. Widersprüche, die nicht auf der Form der Kurvenscheibe beruhen, sind besonders leicht in der Animation erkennbar. Die erzeugten Kurvenscheibengetriebe können in verschiedene Formate exportiert werden. Konfigurationsdateien speichern sämtliche Parameter und Funktionsdefinitionen, die zur Entstehung des Kurvenscheibengetriebes führen, um eine Reproduktion beliebiger Konstruktionen auch an anderen Rechnern zu ermöglichen. Diese Arbeit beinhaltet keine Betrachtung von physikalischen Zusammenhängen, wie Kräfte, Reibung oder Material. Die berechnete Kurvenscheibe ermöglicht dem Abtriebsglied bei ständigem Kontakt, die von der Übertragungsfunktion geforderte Position einzunehmen. Die Zwanglaufführung für das Abtriebsglied ist damit allein jedoch nicht sichergestellt.

Besonderer Wert wurde auf leichte Nachvollziehbarkeit aller Berechnungen gelegt, um Kosten für Implementierung und Wartung einer Software basierend auf den vorgestellten Konzepten gering zu halten. Aus diesem Grund wird häufig von den Empfehlungen der VDI-Richtlinien zugunsten einfacherer, jedoch ebenso funktionaler Methoden

abgewichen. Es wurde ein Prototyp implementiert, um wesentliche Konzepte zu veranschaulichen und die Ergebnisse der Berechnungen zu validieren.
Im Prototyp wurden folgende Ansätze nicht umgesetzt, die für künftige Implementationen wesentliche Verbesserungen mit sich bringen würden:

- Wählbarkeit des Interpolationsverfahren zwischen den Stützpunkten des Kurvenscheibe im Getriebeeditor.
- Das Ändern von Parameter im Getriebeeditor via Drag-Funktionalität.
- Das Optimieren der Kurvenverläufe unter Einbeziehung von definierbaren Toleranzen (s. Abschnitt 2.1.6).
- Das Kombinieren von Getriebetypen zur Bildung formschlüssiger Kurvenscheibengetriebe (s. Abschnitt 4.7.3).
- Stapelverarbeitung von Konfigurationsdateien (s. Abschnitt 4.6.3).

Auf Grundlage der vorgestellten Konzepte kann ein Editor erstellt werden, der sich auch auf andere Kurvenscheibengetriebetypen sowie weitere Verlaufstypen in der Übertragungsfunktion erweitern lässt. Durch die Verwendung plattformunabhängiger Bibliotheken lässt sich der Editor leicht auf andere Betriebssysteme portieren und funktioniert als eigenständige Anwendung. Der Editor bietet Drag-Funktionalitäten für die Übertragungsfunktion, die Auswirkungen in Echtzeit sowohl auf die Übertragungsfunktion und deren Ableitungen als auch auf die Kurvenscheibenkontur selbt visualisieren. Diese Funktionalität wie auch die Möglichkeit zur Veränderung von Parametern während der Animation des erzeugten Getriebes schaffen eine hohe Interaktivität und prädestinieren so den Editor für den Einsatz in der Lehre. Ebenso ist eine Integration in MASP [12] denkbar.

Literaturverzeichnis

[1] Apache Software Foundation: *Xerces-C++*.
http://xerces.apache.org/xerces-c/, . – [Online; Stand 14.03.2008]

[2] Arndt, Jörg: *HUGE-FLOAT package*.
http://www.jjj.de/hfloat/hfloatpage.html, . – [Online; Stand 14.03.2008]

[3] Braune, Reinhard: Bewegungs-Design mit interaktiver Variation von Stützpunkten. In: *VDI-Berichte Nr. 1966*, 2006, S. 47ff

[4] DMG-Lib e.V.: *Digitale Mechanismen- und Getriebebibliothek*.
http://www.dmg-lib.org, . – [Online; Stand 14.03.2008]

[5] FOX-Entwicklerteam: *FOX-Toolkit*. Version 1.6.31.
http://www.fox-toolkit.org/, . – [Online; Stand 14.03.2008]

[6] GiNaC-Entwicklerteam: *GiNaC*. Version 1.4.1.
http://www.ginac.de/, . – [Online; Stand 14.03.2008]

[7] Haible, Bruno ; Kreckel, Richard B.: *CLN*. Version 1.2.0.
http://www.ginac.de/CLN/, . – [Online; Stand 14.03.2008]

[8] Institut für Getriebetechnik & Maschinenbau,
RWTH Aachen: *CADiS*.
http://www.igm.rwth-aachen.de/, . – [Online; Stand 14.03.2008]

[9] Institut für Getriebetechnik,
Leibniz Universität Hannover: *GENESYS*.
http://www.ifg.uni-hannover.de/, . – [Online; Stand 14.03.2008]

[10] MICROSOFT: *Microsoft Visual Studio .NET 2003*. Version 7.1. http://msdn2.microsoft.com/de-de/vstudio/aa700867(en-us).aspx, . – [Online; Stand 14.03.2008]

[11] STOLP, Michael: *Entwicklung eines Software Development Kits für MASP*. Studienarbeit Fachgebiet Graphische Datenverarbeitung TU Ilmenau, 2006

[12] STOLP, Michael: *Modellierung und Export von Mechanismen in der DMG-Lib*. Diplomarbeit Fachgebiet Graphische Datenverarbeitung TU Ilmenau, April 2007

[13] TEXAS INSTRUMENTS: *Derive*. Version 6. 2003. – Computeralgebrasystem, Entwicklung eingestellt

[14] TOMMILA, Mikko: *apfloat*. http://www.apfloat.org/apfloat/, . – [Online; Stand 14.03.2008]

[15] TROLLTECH: *Qt*. Version 4.3.3. http://trolltech.com/products/qt, . – [Online; Stand 14.03.2008]

[16] VDI: *VDI Richtlinie 2143 Blatt 1: Bewegungsgesetze für Kurvengetriebe; Theoretische Grundlagen*. Verein Deutscher Ingenieure VDI; VDI Gesellschaft Konstruktion und Entwicklung; Ausschuß Ebene Kurvnegetriebe, Oktober 1980

[17] VDI: *VDI Richtlinie 2143 Blatt 2: Bewegungsgesetze für Kurvengetriebe; Praktische Anwendung*. Verein Deutscher Ingenieure VDI; VDI-Gesellschaft Entwicklung Konstruktion Vertrieb; Ausschuß Ebene Kurvnegetriebe, Januar 1987

[18] VDI: *VDI Richtlinie 2142 Blatt 1: Auslegung ebener Kurvengetriebe; Grundlagen, Profilberechnung und Konstruktion*. Verein Deutscher Ingenieure VDI; VDI-Gesellschaft Entwicklung Konstruktion Vertrieb; Ausschuß Ebene Kurvengetriebe, Oktober 1994

[19] VOLLMER, Johannes: *Getriebetechnik: Kurvengetriebe*. VEB Verlag Technik, Berlin, 1976

[20] Zapf, Benjamin: *Erstellung von Kurvenscheibengeometrie und Konzipierung eines Kurvenscheibeneditors.* TU-Ilmenau; Hauptseminar Computergraphik, September 2007

Abbildungsverzeichnis

Anhang A

Koeffizientenermittlung mit Derive

Derive [13] ist eine kommerzielle Software für symbolische Berechnungen, die von Texas Instruments entwickelt wurde. Die Entwicklung des Programms wurde mittlerweile eingestellt. Im Rahmen dieser Arbeit wurde die Software vor allem zum Lösen von Gleichungssystemen sowie zur Validierung der Ergebnisse anderer Rechnungen genutzt.

A.1 Polynom 5. Grades

Die allgemeine Form eines Polynoms 5. Grades sowie weitere Anforderungen wurden in Abschnitt 2.1.3 vorgestellt. Für Derive mussten die Variablen (s. Tab. A.1) umbenannt werden. Die daraus resultierenden Forderungen lauten:

- $g^5 \cdot f + g^4 \cdot e + g^3 \cdot d + g^2 \cdot c + g \cdot b + a = h$
- $k^5 \cdot f + k^4 \cdot e + k^3 \cdot d + k^2 \cdot c + k \cdot b + a = l$
- $5 \cdot g^4 \cdot f + 4 \cdot g^3 \cdot e + 3 \cdot g^2 \cdot d + 2 \cdot g \cdot c + b = i$

Variable	x_0	y_0	v_0	a_0	x_1	y_1	v_1	a_1
in Derive	g	h	i	j	k	l	m	n

Tabelle A.1: *Variablen in Derive*

- $5 \cdot k^4 \cdot f + 4 \cdot k^3 \cdot e + 3 \cdot k^2 \cdot d + 2 \cdot k \cdot c + b = m$
- $20 \cdot g^3 \cdot f + 12 \cdot g^2 \cdot e + 6 \cdot g \cdot d + 2 \cdot c = j$
- $20 \cdot k^3 \cdot f + 12 \cdot k^2 \cdot e + 6 \cdot k \cdot d + 2 \cdot c = n$

Um dieses Gleichungssystem in Derive zu lösen ist der folgender Code zu verwenden.

```
SOLVE([
  g^5*f+g^4*e+g^3*d+g^2*c+g*b+a=h,
  k^5*f+k^4*e+k^3*d+k^2*c+k*b+a=l,
  5*g^4*f+4*g^3*e+3*g^2*d+2*g*c+b=i,
  5*k^4*f+4*k^3*e+3*k^2*d+2*k*c+b=m,
  20*g^3*f+12*g^2*e+6*g*d+2*c=j,
  20*k^3*f+12*k^2*e+6*k*d+2*c=n
 ],[a,b,c,d,e,f])
```

Als Lösung für die Koeffizienten erhält man:
$a = k^3 \cdot (g^4 \cdot (j - 3 \cdot n) - g^3 \cdot (8 \cdot i + 2 \cdot j \cdot k - 9 \cdot k \cdot n + 12 \cdot m) + g^2 \cdot (20 \cdot h + 10 \cdot i \cdot k + j \cdot k^2 - 10 \cdot (k^2 \cdot n - 2 \cdot k \cdot m + 2 \cdot l)) - g \cdot k \cdot (10 \cdot h + 2 \cdot i \cdot k - 5 \cdot (k^2 \cdot n - 2 \cdot k \cdot m + 2 \cdot l)) + k^2 \cdot (2 \cdot h - k^2 \cdot n + 2 \cdot (k \cdot m - l)))/(2 \cdot (g^2 - 2 \cdot g \cdot k + k^2)^2 \cdot (k - g)) + (k^2 \cdot n - 2 \cdot k \cdot m + 2 \cdot l)/2$

$b = k^2 \cdot (3 \cdot g^4 \cdot (j - 3 \cdot n) - 4 \cdot g^3 \cdot (6 \cdot i + j \cdot k - 3 \cdot (2 \cdot k \cdot n - 3 \cdot m)) + g^2 \cdot (60 \cdot h + 16 \cdot i \cdot k - j \cdot k^2 - 23 \cdot k^2 \cdot n + 44 \cdot k \cdot m - 60 \cdot l) + 2 \cdot g \cdot k^2 \cdot (5 \cdot i + j \cdot k + 5 \cdot k \cdot n - 5 \cdot m) - 2 \cdot k^3 \cdot (i + k \cdot n - m))/(2 \cdot (g - k) \cdot (g^2 - 2 \cdot g \cdot k + k^2)^2) - k \cdot n + m$

$c = k \cdot (3 \cdot g^4 \cdot (j - 3 \cdot n) + 6 \cdot g^3 \cdot (3 \cdot (k \cdot n - 2 \cdot m) - 4 \cdot i) + 2 \cdot g^2 \cdot (30 \cdot h - 6 \cdot i \cdot k - 4 \cdot j \cdot k^2 - 5 \cdot k^2 \cdot n + 6 \cdot (k \cdot m - 5 \cdot l)) + 2 \cdot g \cdot k \cdot (30 \cdot h + 18 \cdot i \cdot k + 2 \cdot j \cdot k^2 + k^2 \cdot n + 12 \cdot k \cdot m - 30 \cdot l) + k^4 \cdot (j - n))/(2 \cdot (g^2 - 2 \cdot g \cdot k + k^2)^2 \cdot (k - g)) + n/2$

$d = (g^4 \cdot (j - 3 \cdot n) - 4 \cdot g^3 \cdot (2 \cdot i - j \cdot k + 3 \cdot m) + 4 \cdot g^2 \cdot (5 \cdot h - 8 \cdot i \cdot k - 2 \cdot j \cdot k^2 + 2 \cdot k^2 \cdot n - 7 \cdot k \cdot m - 5 \cdot l) + 4 \cdot g \cdot k \cdot (20 \cdot h + 7 \cdot i \cdot k - k^2 \cdot n + 4 \cdot (2 \cdot k \cdot m - 5 \cdot l)) + k^2 \cdot (20 \cdot h + 12 \cdot i \cdot k + 3 \cdot j \cdot k^2 - k^2 \cdot n + 4 \cdot (2 \cdot k \cdot m - 5 \cdot l)))/(2 \cdot (g - k) \cdot (g^2 - 2 \cdot g \cdot k + k^2)^2)$

$e = (g^3 \cdot (2 \cdot j - 3 \cdot n) - g^2 \cdot (14 \cdot i + j \cdot k - 4 \cdot (k \cdot n - 4 \cdot m)) + g \cdot (30 \cdot h - 2 \cdot i \cdot k - 4 \cdot j \cdot k^2 + k^2 \cdot n + 2 \cdot k \cdot m - 30 \cdot l) + k \cdot (30 \cdot h + 16 \cdot i \cdot k + 3 \cdot j \cdot k^2 - 2 \cdot (k^2 \cdot n - 7 \cdot k \cdot m + 15 \cdot l)))/(2 \cdot (g^2 - 2 \cdot g \cdot k + k^2)^2 \cdot (k - g))$

$f = (g^2 \cdot (j - n) - 2 \cdot g \cdot (3 \cdot i + j \cdot k - k \cdot n + 3 \cdot m) + 12 \cdot h + 6 \cdot i \cdot k + j \cdot k^2 - k^2 \cdot n + 6 \cdot (k \cdot m - 2 \cdot l))/(2 \cdot (g^5 - 5 \cdot g^4 \cdot k + 10 \cdot g^3 \cdot k^2 - 10 \cdot g^2 \cdot k^3 + 5 \cdot g \cdot k^4 - k^5))$

Für die Benutzung dieser Lösungen in einem Programm empfehlen sich einige Optimierungen. Als gutes Beispiel hierfür dient der Nenner in den Gleichungen. Dieser lässt sich zu $2 \cdot (g - k)^5$ bzw. $-2 \cdot (g - k)^5$ umformen. Diese Berechnung muss nur einmal erfolgen und das Ergebnis kann in den Berechnungen aller sechs Koeffizienten genutzt werden. Obendrein kann eine solche Umformung auch die Rechengenauigkeit erhöhen (s. Abschnitt 4.1.5).

Anhang B

Beispiele

Um die einzelnen Schritte besser nachvollziehen zu können, sollen in diesem Kapitel einige Beispiele aufgeführt werden.

B.1 Rollenstößel mit zwei Rasten

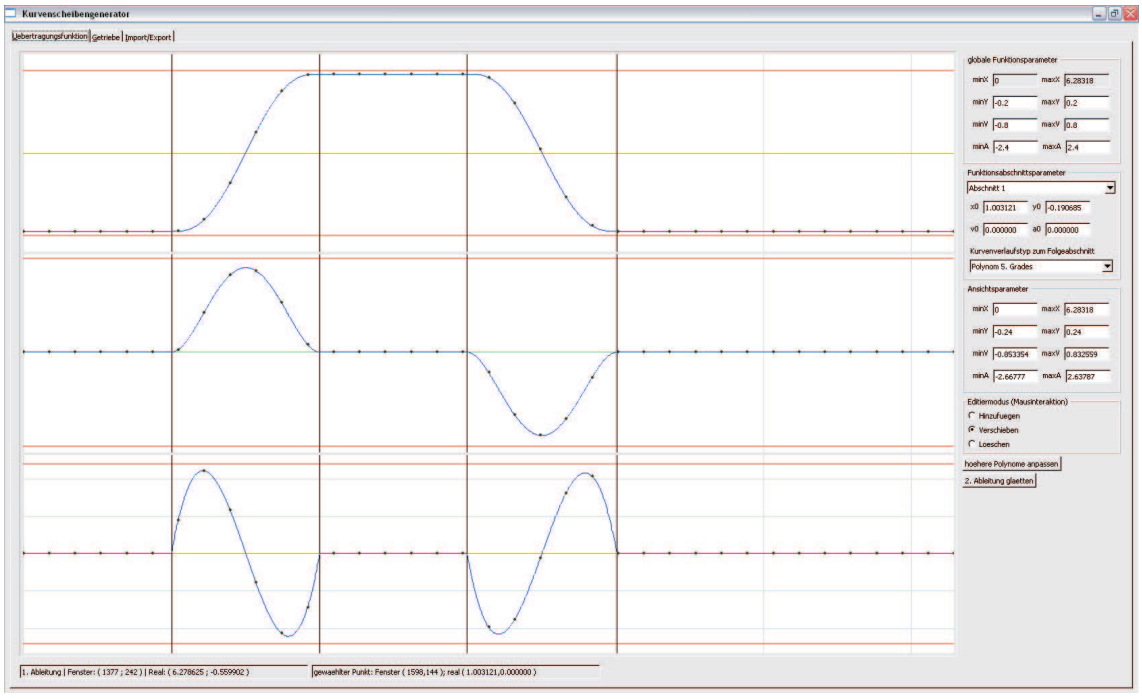


Abbildung B.1: *Übertragungsfunktion mit zwei Rastabschnitten*

Abbildung B.1 ist eine einfache Übertragungsfunktion mit vier Funktionsabschnitten dargestellt. Es handelt sich dabei um zwei Rastabschnitte, die jeweils durch ein Polynom 5. Grades verbunden sind. 36 Abtastwerte sind markiert und sollen für dieses Beispiel ausreichend sein. Mit dieser Funktion könnte beispielsweise ein einfacher Schalter realisiert werden.

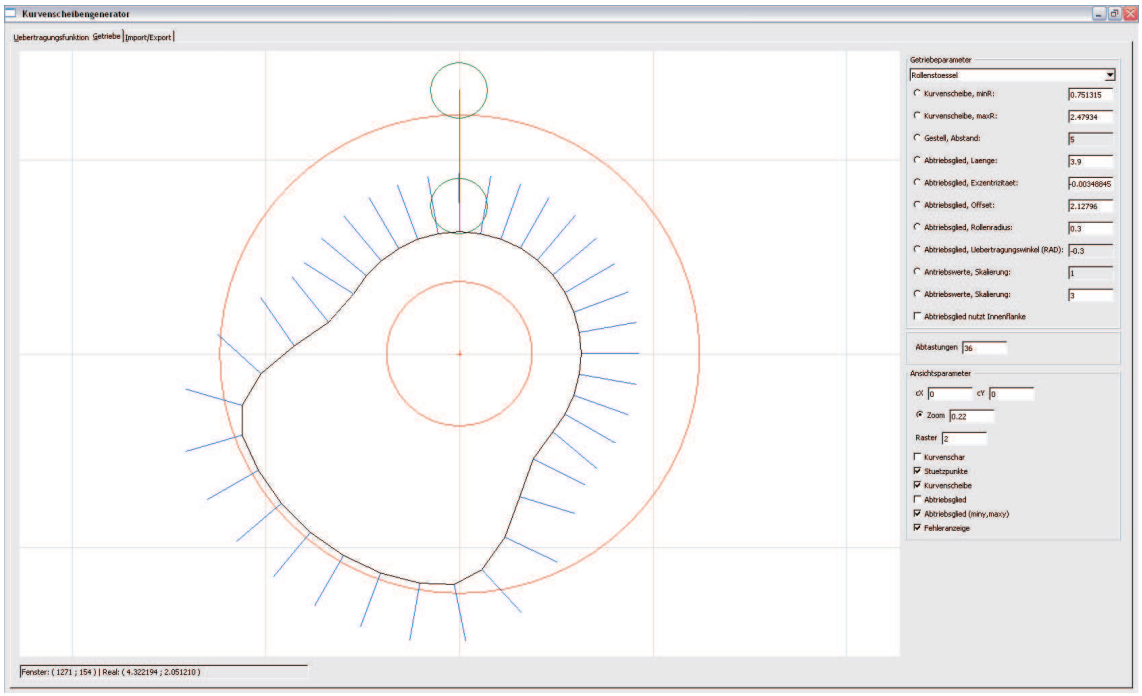


Abbildung B.2: *Kurvenscheibe und Rollenstößel mit geringer Exzentrizität*

Abbildung B.2 zeigt einen Rollenstößel mit geringer Exzentrizität und die Kurvenscheibe, die die zugehörige Übertragungsfunktion aus Abb. B.1 realisiert.
Die Konfigurationsdatei, die zu dieser Konfiguration führte, beinhaltet folgende Daten:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<CamDiscGenerator>

  <FunctionParam
   maxA="2.400000" maxV="0.800000" maxX="6.283185" maxY="0.200000"
   minA="-2.400000" minV="-0.800000" minX="0.000000" minY="-0.200000"
  />
```

```
 <MechParam
  excentr="-0.003488"
  innerFlank="0"
  lengthA="5.000000"
  lengthB="3.900000"
  maxRadius="2.479339"
  minRadius="0.751315"
  offset="2.127956"
  rollRadius="0.300000"
  samples="36"
  scaleIn="1.000000"
  scaleOut="3.000000"
  transAngle="-0.300000"
  type="RolledLifter"
 />

 <ViewParam/>

 <Function>
   <Section a="0.000000" type="CDD_FS_POLY5"
    v="0.000000" x="1.003121" y="-0.190685"/>
   <Section a="0.000000" type="CDD_FS_CONST"
v="0.000000" x="2.001682" y="0.192329"/>
   <Section a="0.000000" type="CDD_FS_POLY5"
v="0.000000" x="2.995684" y="0.192329"/>
   <Section a="0.000000" type="CDD_FS_CONST"
v="0.000000" x="4.007924" y="-0.190685"/>
 </Function>

</CamDiscGenerator>
```

Das Getriebe als MASP-Datei [12] exportiert beinhaltet folgende Daten:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<Mechanism
 xmlns="http://www.dmg-lib.org/mechanism"
 dmglib_MecID="0"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.dmg-lib.org/mechanism
  http://dmg-lib.org/dmglib/schemas/mechanism.xsd">

  <ComponentSpaces xmlns="http://www.dmg-lib.org/componentspaces">
    <ComponentSpace id="cspace1" type="planar">
      <Positions>
        <Position id="pos1" x="0" y="0"/>
        <Position id="pos2" x="-0.00281568" y="1.2559"/>
        <Position id="pos3" x="-0.220858" y="1.23633"/>
        <Position id="pos4" x="-0.43219" y="1.1792"/>
        <Position id="pos5" x="-0.63039" y="1.08624"/>
        <Position id="pos6" x="-0.809435" y="0.960267"/>
        <Position id="pos7" x="-0.963887" y="0.805122"/>
        <Position id="pos8" x="-1.09595" y="0.615829"/>
        <Position id="pos9" x="-1.35593" y="0.322912"/>
        <Position id="pos10" x="-1.71129" y="0.0797497"/>
        <Position id="pos11" x="-2.05439" y="-0.205536"/>
        <Position id="pos12" x="-2.24946" y="-0.533397"/>
        <Position id="pos13" x="-2.24692" y="-0.844155"/>
        <Position id="pos14" x="-2.08119" y="-1.20516"/>
        <Position id="pos15" x="-1.8403" y="-1.54824"/>
        <Position id="pos16" x="-1.54349" y="-1.84429"/>
        <Position id="pos17" x="-1.19979" y="-2.08429"/>
        <Position id="pos18" x="-0.819625" y="-2.26097"/>
        <Position id="pos19" x="-0.41456" y="-2.36895"/>
        <Position id="pos20" x="-0.053894" y="-2.38308"/>
        <Position id="pos21" x="0.232776" y="-2.23254"/>
        <Position id="pos22" x="0.468868" y="-1.89471"/>
```

```
    <Position id="pos23" x="0.618475" y="-1.47695"/>
    <Position id="pos24" x="0.762981" y="-1.08154"/>
    <Position id="pos25" x="0.963887" y="-0.805122"/>
    <Position id="pos26" x="1.08905" y="-0.625513"/>
    <Position id="pos27" x="1.18113" y="-0.426898"/>
    <Position id="pos28" x="1.23731" y="-0.215313"/>
    <Position id="pos29" x="1.2559" y="0.00281539"/>
    <Position id="pos30" x="1.23633" y="0.220858"/>
    <Position id="pos31" x="1.1792" y="0.43219"/>
    <Position id="pos32" x="1.08624" y="0.630389"/>
    <Position id="pos33" x="0.960267" y="0.809435"/>
    <Position id="pos34" x="0.805122" y="0.963887"/>
    <Position id="pos35" x="0.625513" y="1.08905"/>
    <Position id="pos36" x="0.426898" y="1.18113"/>
    <Position id="pos37" x="0.215313" y="1.23731"/>
    <Position id="pos38" x="-0.00348845" y="2.72796"/>
    <Position id="pos39" x="-0.00348845" y="1.52796"/>
    <Position id="pos40" x="-0.00348845" y="1.5559"/>
    <Position id="pos41" x="-0.00348845" y="1.5559"/>
    <Position id="pos42" x="-0.00348845" y="1.5559"/>
  </Positions>
  <Scalars>
    <Scalar id="scalar1" value="0"/>
    <Scalar id="scalar2" value="0"/>
    <Scalar id="scalar3" value="-1.5708"/>
    <Scalar id="scalar4" value="0.00348845"/>
    <Scalar id="scalar5" value="1.2"/>
    <Scalar id="scalar6" value="1.5559"/>
    <Scalar id="scalar7" value="0"/>
    <Scalar id="scalar8" value="0.3"/>
    <Scalar id="scalar9" value="1.5559"/>
    <Scalar id="scalar10" value="0"/>
    <Scalar id="scalar11" value="1.5559"/>
```

```
    <Scalar id="scalar12" value="0"/>
  </Scalars>
  <Elements>
    <Bar fixed="true" id="bar2">
      <Point fixed="false" id="bar2.pt1" posRef="pos38"/>
      <Point fixed="false" id="bar2.pt2" posRef="pos39"/>
      <Line fixed="true" id="bar2.line">
        <Param fixed="false" id="bar2.line.dir" scalarRef="scalar3"/>
        <Param fixed="false" id="bar2.line.offs" scalarRef="scalar4"/>
      </Line>
      <Param fixed="false" id="bar2.length" scalarRef="scalar5"/>
    </Bar>
    <Wheel fixed="false" id="wheel3">
      <Point fixed="false" id="wheel3.pt1" posRef="pos40"/>
      <Line fixed="false" id="wheel3.line">
        <Param fixed="false" id="wheel3.line.dir" scalarRef="scalar7"/>
        <Param fixed="false" id="wheel3.line.offs" scalarRef="scalar6"/>
      </Line>
      <Param fixed="false" id="wheel3.radius" scalarRef="scalar8"/>
    </Wheel>
    <Camdisc fixed="false" id="Kurvenscheibe" xsi:type="CamdiscPolar">
      <Point fixed="false" id="Kurvenscheibe.pt1" posRef="pos1"/>
      <Point fixed="false" id="Kurvenscheibe.pt2" posRef="pos2"/>
      <Point fixed="false" id="Kurvenscheibe.pt3" posRef="pos3"/>
      <Point fixed="false" id="Kurvenscheibe.pt4" posRef="pos4"/>
      <Point fixed="false" id="Kurvenscheibe.pt5" posRef="pos5"/>
      <Point fixed="false" id="Kurvenscheibe.pt6" posRef="pos6"/>
      <Point fixed="false" id="Kurvenscheibe.pt7" posRef="pos7"/>
      <Point fixed="false" id="Kurvenscheibe.pt8" posRef="pos8"/>
      <Point fixed="false" id="Kurvenscheibe.pt9" posRef="pos9"/>
      <Point fixed="false" id="Kurvenscheibe.pt10" posRef="pos10"/>
      <Point fixed="false" id="Kurvenscheibe.pt11" posRef="pos11"/>
      <Point fixed="false" id="Kurvenscheibe.pt12" posRef="pos12"/>
```

```
      <Point fixed="false" id="Kurvenscheibe.pt13" posRef="pos13"/>
      <Point fixed="false" id="Kurvenscheibe.pt14" posRef="pos14"/>
      <Point fixed="false" id="Kurvenscheibe.pt15" posRef="pos15"/>
      <Point fixed="false" id="Kurvenscheibe.pt16" posRef="pos16"/>
      <Point fixed="false" id="Kurvenscheibe.pt17" posRef="pos17"/>
      <Point fixed="false" id="Kurvenscheibe.pt18" posRef="pos18"/>
      <Point fixed="false" id="Kurvenscheibe.pt19" posRef="pos19"/>
      <Point fixed="false" id="Kurvenscheibe.pt20" posRef="pos20"/>
      <Point fixed="false" id="Kurvenscheibe.pt21" posRef="pos21"/>
      <Point fixed="false" id="Kurvenscheibe.pt22" posRef="pos22"/>
      <Point fixed="false" id="Kurvenscheibe.pt23" posRef="pos23"/>
      <Point fixed="false" id="Kurvenscheibe.pt24" posRef="pos24"/>
      <Point fixed="false" id="Kurvenscheibe.pt25" posRef="pos25"/>
      <Point fixed="false" id="Kurvenscheibe.pt26" posRef="pos26"/>
      <Point fixed="false" id="Kurvenscheibe.pt27" posRef="pos27"/>
      <Point fixed="false" id="Kurvenscheibe.pt28" posRef="pos28"/>
      <Point fixed="false" id="Kurvenscheibe.pt29" posRef="pos29"/>
      <Point fixed="false" id="Kurvenscheibe.pt30" posRef="pos30"/>
      <Point fixed="false" id="Kurvenscheibe.pt31" posRef="pos31"/>
      <Point fixed="false" id="Kurvenscheibe.pt32" posRef="pos32"/>
      <Point fixed="false" id="Kurvenscheibe.pt33" posRef="pos33"/>
      <Point fixed="false" id="Kurvenscheibe.pt34" posRef="pos34"/>
      <Point fixed="false" id="Kurvenscheibe.pt35" posRef="pos35"/>
      <Point fixed="false" id="Kurvenscheibe.pt36" posRef="pos36"/>
      <Point fixed="false" id="Kurvenscheibe.pt37" posRef="pos37"/>
      <Line fixed="false" id="Kurvenscheibe.line">
        <Param fixed="false" id="Kurvenscheibe.line.dir" scalarRef="scalar2"/>
        <Param fixed="false" id="Kurvenscheibe.line.offs" scalarRef="scalar1"/>
      </Line>
    </Camdisc>
  </Elements>
  <Couplings>
    <SwivelJoint fixed="false" id="swj2">
```

```
          <Point fixed="false" id="swj2.pt" posRef="pos42"/>
          <Line fixed="false" id="swj2.line">
            <Param fixed="false" id="swj2.line.dir" scalarRef="scalar12"/>
            <Param fixed="false" id="swj2.line.offs" scalarRef="scalar11"/>
          </Line>
        </SwivelJoint>
        <SlidingJoint fixed="false" id="slj1">
          <Point fixed="false" id="slj1.pt" posRef="pos41"/>
          <Line fixed="false" id="slj1.line">
            <Param fixed="false" id="slj1.line.dir" scalarRef="scalar10"/>
            <Param fixed="false" id="slj1.line.offs" scalarRef="scalar9"/>
          </Line>
        </SlidingJoint>
      </Couplings>
      <Connections>
        <PointConnection couplingPointRef="swj2.pt"
 elementPointRef="wheel3.pt1" id="pointcon1"/>
        <LineConnection couplingLineRef="slj1.line"
 elementLineRef="bar2.line" id="linecon2"/>
        <CouplingConnection coupling1Ref="slj1"
 coupling2Ref="swj2" id="couplingcon3"
 role="SlidingSwivelJoint"/>
      </Connections>
    </ComponentSpace>
  </ComponentSpaces>

</Mechanism>
```

Das in Abb. B.3 dargestellte Getriebe erfüllt dieselbe Übertragungsfunktion, wie das in Abb. B.2. Lediglich Offset und Exzentrizität des Abtriebsglieds wurden verändert. Die im Verhältnis übermäßige Exzentrizität dürfte in der Praxis jedoch zu Problemen führen, da die Kraft, die die Kurvenscheibe auf das Abtriebsglied ausübt, in eine ungünstige Richtung zeigt.

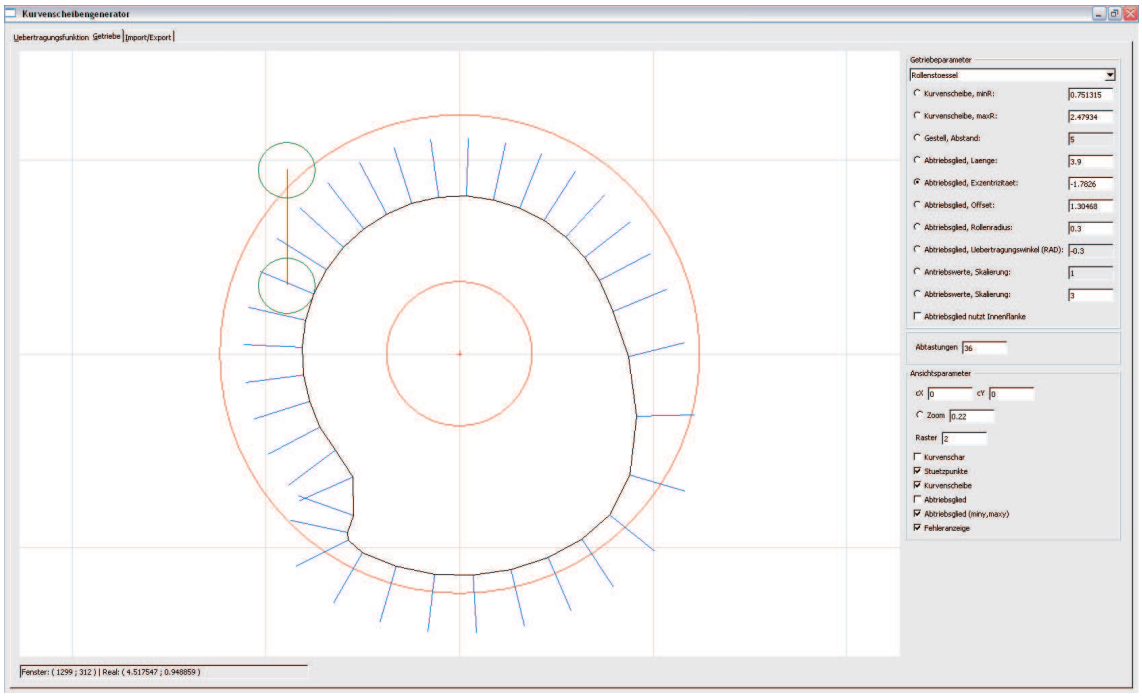


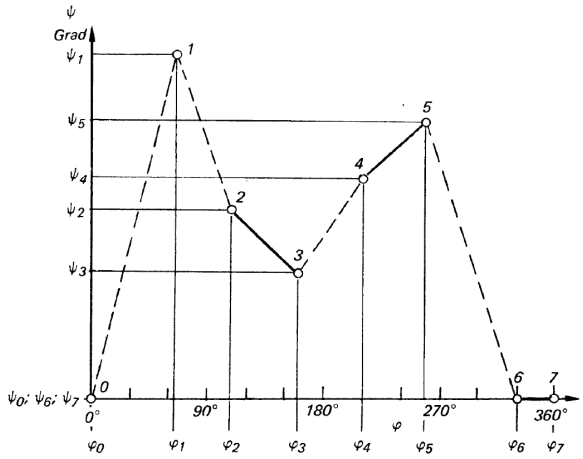
Abbildung B.3: *Kurvenscheibe und Rollenstößel mit hoher Exzentrizität*

B.2 Übertragungsfunktion aus Bewegungsplan

In der VDI-Richtlinie 2143 Blatt 2 [17, S. 38] findet sich ein Beispiel für die Erstellung eines Bewegungsdiagramms aus einem Bewegungsplan (s. Abb. B.4), welches hier mit dem vorgestellten Editor nachvollzogen wird.

Zunächst wird sichergestellt, dass für die Eingabe der Übertragungsfunktion angemessene Bereiche im Editor gewählt sind. Gradangaben im Bewegungsplan sind hier in das Bogenmaß umzurechnen. Hierbei kommt folgende Formel zur Anwendung: $x_{Rad} = x_{Deg}/360 * 2 * \pi$. Die Ergebnisse sind in Tabelle B.1 aufgeführt.

Jetzt werden die Abschnittsgrenzen mit Hilfe der Maus hinzugefügt. Hier empfiehlt es sich die Eingaben im oberen Fenster vorzunehmen und die Opition zu wählen, dass nicht gesetzte Parameter mit 0 initialisiert werden. Nach Eingabe der Koordinaten mit der Maus im oberen Fenster ergibt sich der Verlauf entsprechend Abb. B.5. Die nicht gesetzten Parameter werden anschließend über das vertikale Verschieben der Parameter in den anderen beiden Fenstern angepasst. Bereits jetzt kommt das Ergebnis (Abb. B.6)



Punkt	φ in Grad	ψ in Grad	Bewegungsaufgabe	ψ′ in rad/rad $= \frac{\omega_b}{\omega_a}$	ψ″ in rad/rad^2 $= \frac{\alpha_b}{\omega_a^2}$
0	0	0	R	0	0
1	65	55	U	0	(-3,481227)
2	110	(29,549597)	G	(-0,190986)	0
3	160	20	G	(-0,190986)	0
4	210	35	G	(0,190986)	0
5	260	(44,549297)	G	(0,190986)	0
6	340	0	R	0	0
7	360	0	R	0	0

Abbildung B.4: *Bewegungsplan aus VDI-Richtlinie*

der Abbildung aus den VDI-Richtlinien B.7 recht nahe.

Dabei ist zu beachten, dass der Systematik der Bewegungsaufgaben bislang keine Beachtung geschenkt wurde. Die Verlaufstypen der Funktionsabschnitte sind standardgemäß Polynome 5. Grades und somit automatisch stoß- und ruckfrei. Weitere Optimierungen könnnen vorgenommen werden, indem die Parameter der Abschnittsgrenzen über Tastatureingaben korrigiert werden (Die Eingabe der mit der Maus lässt keine exakte Parametrisierung zu). Gemäß des Bewegungsplans können Rastabschnitten das Polynom 0. Grades und Geradenabschnitten das Polynome 1. Grades zugewiesen werden. Das Ergebnis nach einer abschließenden automatischen Parameterkorrektur, ist in Abbildung B.8 zu sehen. Die automatische Parameterkorrektur hat den Beschleunigungswert von Punkt 1 von -3.481227 auf -4.486472 verändert, um den Kurvenverlauf der 2. Ableitung nach Möglichkeit zu glätten. Abgesehen davon resultieren Unterschiede zu Abb. B.7 einzig daraus, dass dort anstelle des Polynom 5. Grades andere ebenso zulässige Bewegungsgesetze gewählt wurden.

Deg	*Rad*	*Deg*	*Rad*
0	0, 000000	210	3, 665191
20	0, 349066	260	4, 537856
35	0, 610865	340	5, 934119
55	0, 959931	360	6, 283185
65	1, 134464	29, 549597	0, 515738
110	1, 919862	44, 549297	0, 777532
160	2, 792527		

Tabelle B.1: *Umrechnung von Grad ins Bogenmaß*

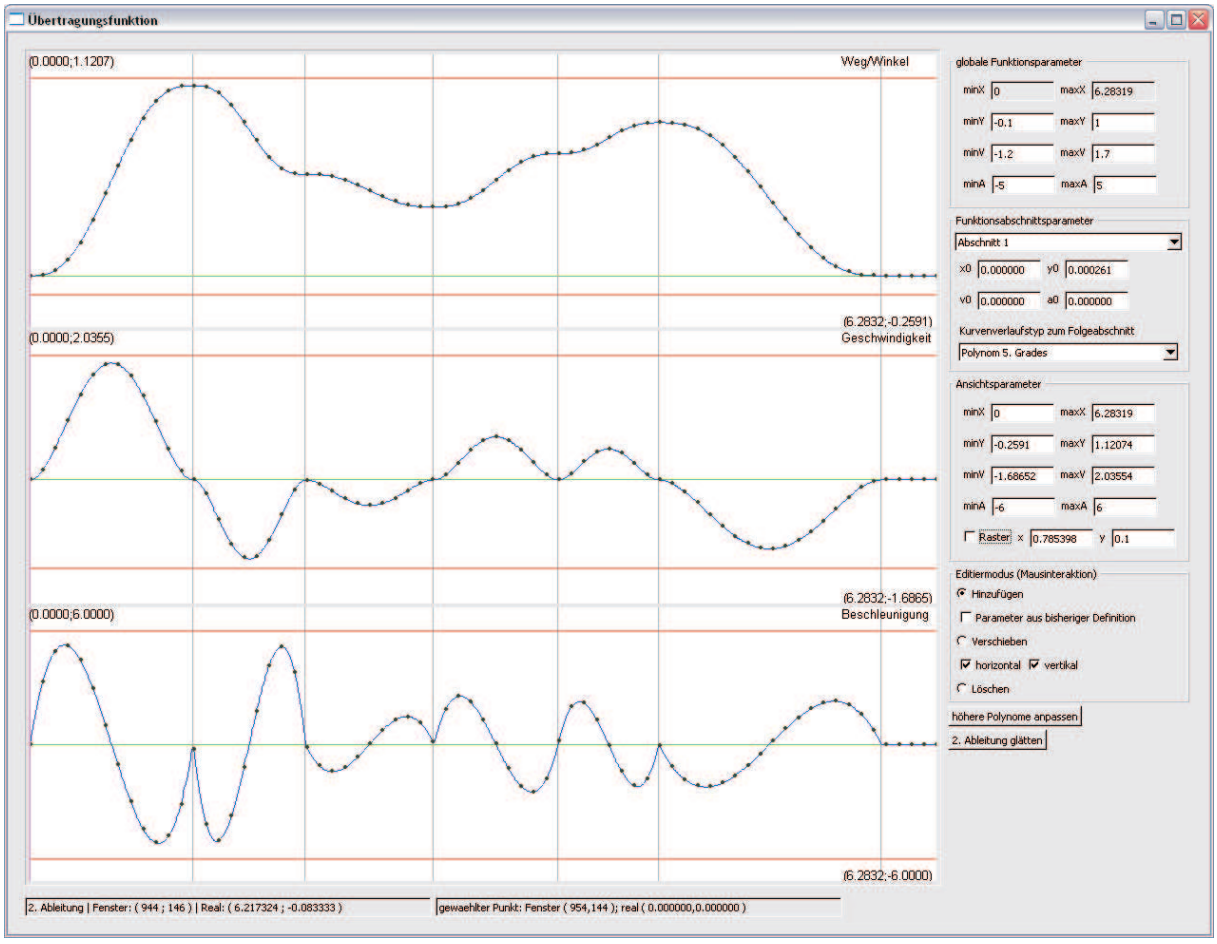


Abbildung B.5: *Kurvenverlauf nach Hinzufügen der Abschnittsgrenzen im oberen Fenster*

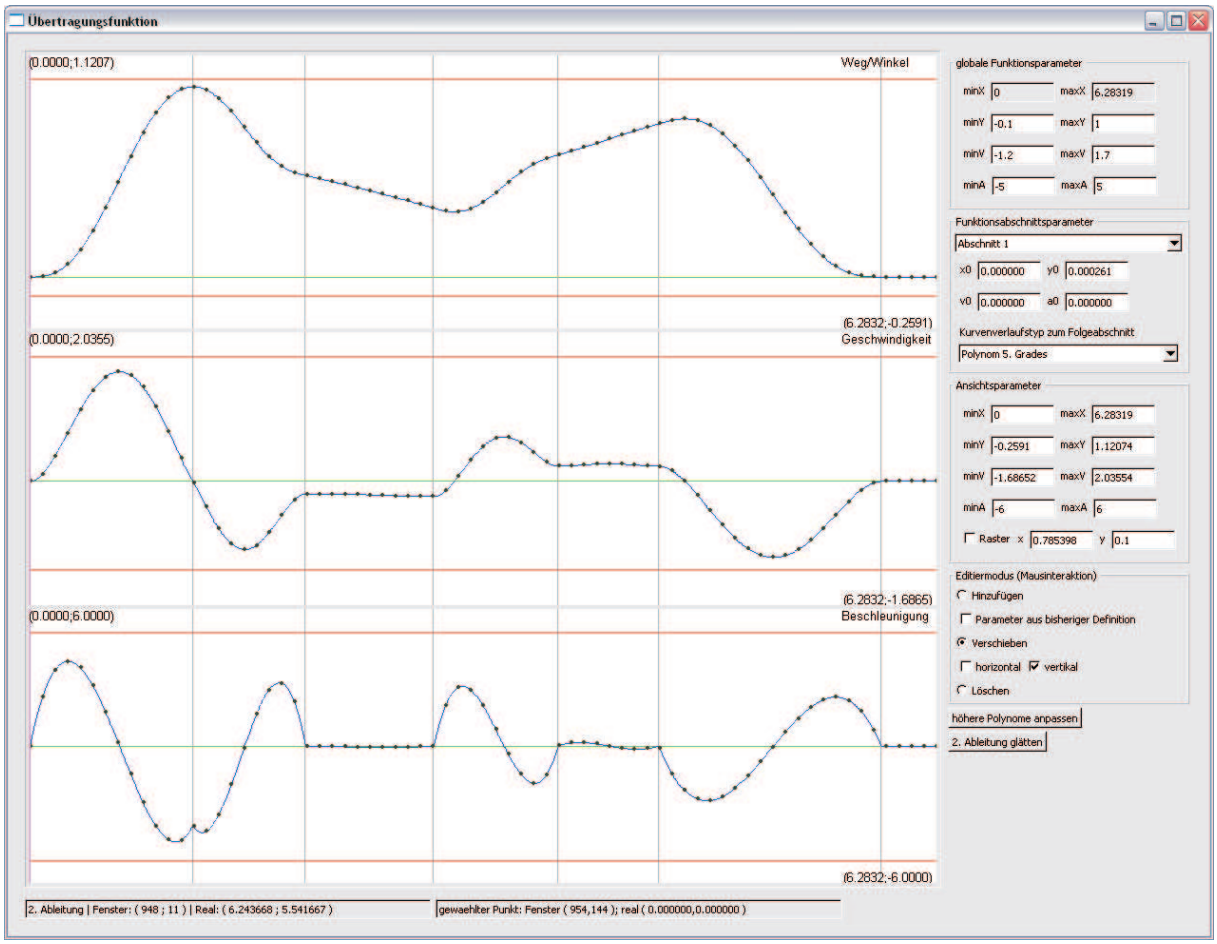


Abbildung B.6: *Kurvenverlauf nach Eingabe des Bewegungsplans*

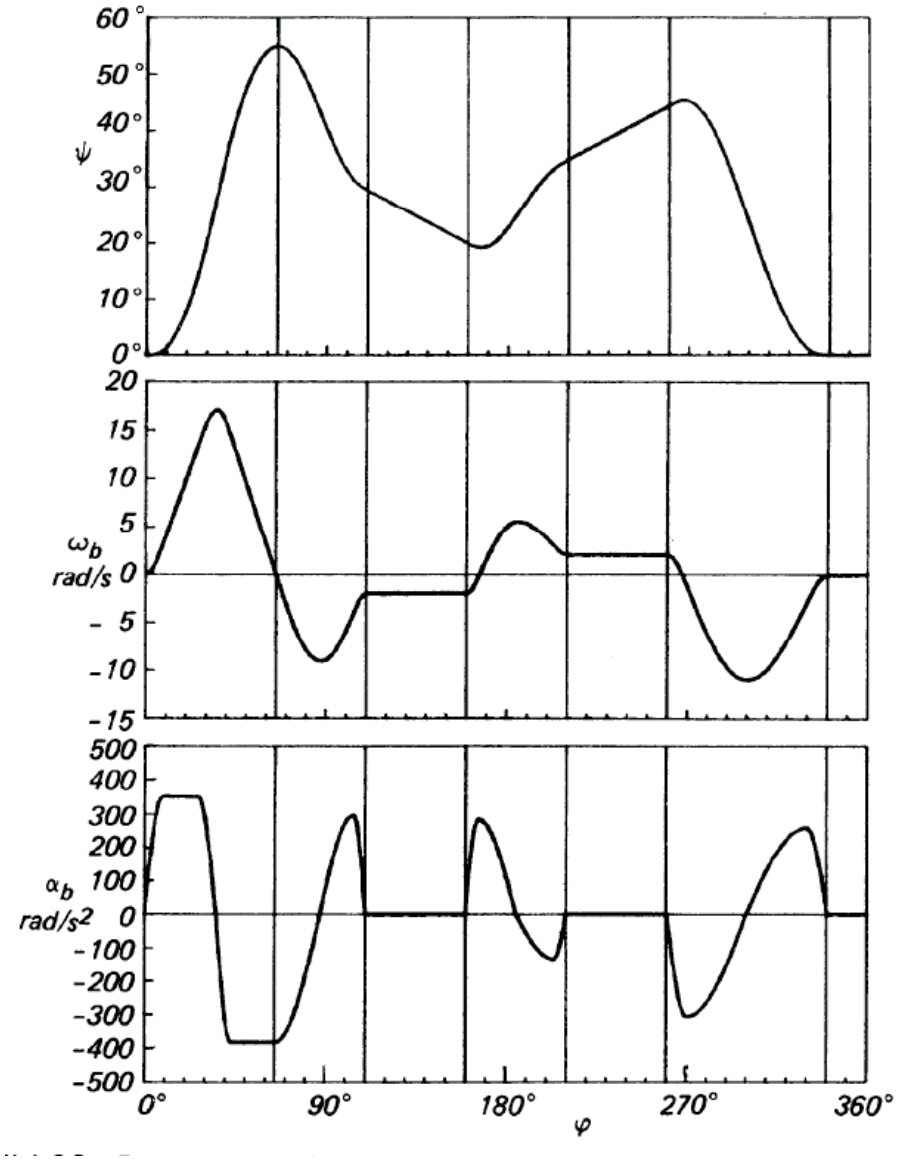


Abbildung B.7: *Bewegungsdiagramm aus VDI-Richtlinie*

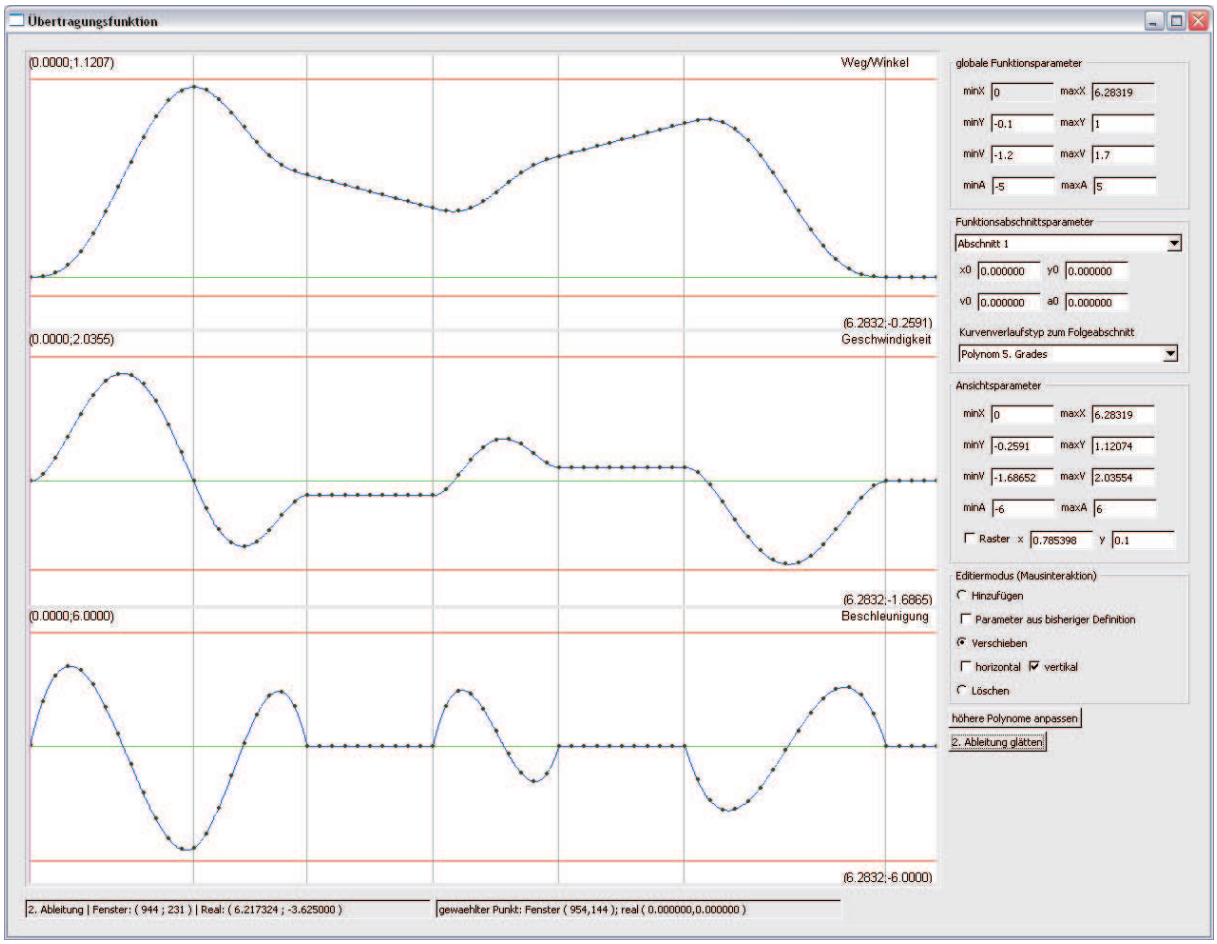


Abbildung B.8: *Kurvenverlauf nach Optimierung*

Thesen

- Der Verzicht auf Nutzung normalisierter Funktionen erspart häufige Skalierungen inklusive ihrer Sonderfälle.
- Der Verzicht auf die Systematik der Bewegungsaufgaben reduziert die Zahl der zu betrachtenden Fälle erheblich.
- Numerische Berührpunktermittlung basierend auf einfachen geometrischen Zusammenhängen erlaubt bei hinreichender Detailgenauigkeit den Verzicht auf zusätzliche Bibliotheken und verringert den Implementations- und Wartungsaufwand.
- Das vorgestellte iterative Verfahren zur automatischen Parameteranpassung erlaubt die global gleichmäßige Glättung des Beschleunigungsverlaufs bei Verwendung von Polynomen 5. Grades.
- Durch hohe Abtastzahlen erhöht sich die Zahl der zu verarbeitenden Daten, jedoch kann nur so eine hinreichende Nachbildung der definierten Übertragungsfunktion gewährleistet werden. Ebenso ist eine hohe Abtastzahl für eine hinreichend genaue Definition der Kurvenscheibenkontur essentiell.
- Die Haltung von Parametern, die für den aktuellen Getriebetyp oder Funktionsabschnitt nicht benötigt werden, erlaubt einen schnellen Wechsel zu anderen Typen und erhöht so die Editierbarkeit.
- Parameteränderungen während der Animation und Drag-Funktionalitäten mit synchroner Darstellung der Auswirkungen in anderen Ansichten unterstützen sehr stark die Interaktivität.

Ilmenau, 14.03.2008

Jens Storz